

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

CRÉATION DE CONNAISSANCES DANS LE LOGICIEL LIBRE :
UNE ANALYSE DE LA COMMUNAUTÉ UBUNTU

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DU MBA-RECHERCHE EN MANAGEMENT ET TECHNOLOGIE

PAR
GUILLAUME BLUM

JANVIER 2007

UNIVERSITÉ DU QUÉBEC À MONTRÉAL
Service des bibliothèques

Avertissement

La diffusion de ce mémoire se fait dans le respect des droits de son auteur, qui a signé le formulaire *Autorisation de reproduire et de diffuser un travail de recherche de cycles supérieurs* (SDU-522 – Rév.01-2006). Cette autorisation stipule que «conformément à l'article 11 du Règlement no 8 des études de cycles supérieurs, [l'auteur] concède à l'Université du Québec à Montréal une licence non exclusive d'utilisation et de publication de la totalité ou d'une partie importante de [son] travail de recherche pour des fins pédagogiques et non commerciales. Plus précisément, [l'auteur] autorise l'Université du Québec à Montréal à reproduire, diffuser, prêter, distribuer ou vendre des copies de [son] travail de recherche à des fins non commerciales sur quelque support que ce soit, y compris l'Internet. Cette licence et cette autorisation n'entraînent pas une renonciation de [la] part [de l'auteur] à [ses] droits moraux ni à [ses] droits de propriété intellectuelle. Sauf entente contraire, [l'auteur] conserve la liberté de diffuser et de commercialiser ou non ce travail dont [il] possède un exemplaire.»

REMERCIEMENTS

Bien que je sois l'auteur de ce mémoire, il m'aurait été difficile, voire impossible de l'écrire sans l'aide d'un certain nombre de personnes que je tiens à remercier ici pour leurs contributions indirectes à cette œuvre.

Mes premiers remerciements vont bien évidemment au professeur Mehran Ebrahimi, mon directeur, qui a su me laisser voler de mes propres ailes dans le monde de la recherche, tout en sachant me réorienter lorsque je m'écartais de la route. Merci aussi pour ces explications et ces clarifications données, souvent demandées plusieurs fois et auxquelles j'ai toujours eu des réponses éclairées. Merci aussi de m'avoir soutenu dans la voie vers le doctorat, et avoir accepté de me supporter pour encore plusieurs années.

Merci aux professeurs Anne-Laure Saives et Albert Lejeune pour leurs judicieuses remarques, qui ont contribué à la rédaction finale de ce mémoire.

Merci aux différents contributeurs qui m'ont aidé pour la recherche, leurs collaborations m'ont été fort précieuses, et m'ont évité de m'égarer dans ce vaste monde qu'est le logiciel libre.

Merci à Monsieur William-David Holford, pour ses précieux conseils et corrections, gratifiant ce mémoire d'un *abstract* valide et sans fautes dans la langue de Shakespeare.

Merci à tous les membres de la communauté Ubuntu et plus généralement aux communautés du libre, pour tous ces beaux logiciels qu'ils produisent, et sans lesquels mon quotidien informatique (mais je ne suis pas le seul) serait un enfer.

Merci au professeur Jorge Niosi pour son cours d'innovation, qui m'a permis de mieux cerner cette notion, qu'il y a cinq ans encore je prenais pour un subterfuge de gestionnaires pour se justifier de ne pas inventer. Que de chemin parcouru!

Merci à la professeure Renée Bédard d'avoir su me montrer la voie à suivre et avoir su trouver les mots et les actes justes qui m'ont fait me diriger vers la recherche en management. Sans cela, mes trajectoires professionnelle et personnelle auraient été tout autres.

Merci au professeur Omar Aktouf, pour m'avoir tant apporté intellectuellement, et m'avoir tant appris. Merci d'avoir su m'ouvrir au monde de la recherche, et avoir su le premier m'orienter dans une perspective humaniste. Merci au professeur et merci à l'homme.

Merci au groupe GEIRSO, pour m'avoir permis d'utiliser en dehors des heures de travail ses locaux et le matériel nécessaire à la rédaction de ce mémoire.

Merci à tous mes amis qui ont su me supporter (ou ne m'ont pas fait savoir qu'ils ne me supportaient plus!) pendant la phase de travail sur le terrain et la période de rédaction. Il leur fallait un certain courage.

Merci à madame Charlotte Blanche, ma collègue d'étude et de travail tout au long du MBA-Recherche, qui m'a offert nombre de fenêtres de détente, de socialisation au sens de Nonaka, me donnant plusieurs pistes de réflexion qui ont su m'être utile.

Un grand merci aussi à mes parents, qui par leur soutien moral et matériel m'ont permis de venir étudier au Québec sans soucis, pour avoir su toujours me soutenir, et m'ont encouragé à trouver la voie dans laquelle je m'épanouirai.

Enfin, merci, merci beaucoup à ma conjointe Mathilde pour ses relectures, ses conseils, ses encouragements, son soutien tout au long de ce mémoire. Même si cette aide a été réciproque, elle me fut vraiment très précieuse et appréciée à sa juste mesure.

TABLE DES MATIÈRES

LISTE DES FIGURES	VIII
LISTE DES TABLEAUX	IX
LISTE DES ABRÉVIATIONS, DES SIGNES ET DES ACRONYMES	X
LISTE DES ABRÉVIATIONS, DES SIGNES ET DES ACRONYMES	X
RÉSUMÉ & ABSTRACT	XI
RÈGLES DE NOTATION PARTICULIÈRES UTILISÉES	XIII
<i>Distinction des citations suivant leurs origines</i>	<i>xiii</i>
<i>Identification des répondants</i>	<i>xiii</i>
INTRODUCTION.....	1
<i>Intérêt de la recherche.....</i>	<i>2</i>
<i>Problématique de recherche.....</i>	<i>3</i>
<i>Cheminement du mémoire</i>	<i>3</i>
CHAPITRE I	5
REVUE DE LITTÉRATURE – GESTION DE LA CONNAISSANCE	5
<i>1.1 L'économie de la connaissance</i>	<i>5</i>
1.1.1 Les TIC et nouvelle économie ?.....	5
1.1.2 Le savoir dans l'économie	8
1.1.3 L'importance de l'innovation.....	10
1.1.4 La protection de la propriété intellectuelle.....	11
1.1.5 Une nouvelle forme d'organisation : l'organisation en réseau	15
<i>1.2 La connaissance</i>	<i>16</i>
1.2.1 L'apprentissage	20
<i>1.3 La gestion de la connaissance</i>	<i>20</i>
1.3.1 Les origines.....	20
1.3.2 La gestion des connaissances	21
1.3.2.1 L'école technologique	22
1.3.2.2 L'école organisationnelle.....	23
1.3.3 Les conditions et phases de la création de connaissance.....	25

1.3.4 La notion de « Ba ».....	27
<i>1.4 La théorie de la firme créatrice de connaissance.....</i>	<i>28</i>
<i>1.5 Les communautés de pratique.....</i>	<i>31</i>
1.5.1 Les autres formes de collectifs.....	31
1.5.2 La communauté de pratique : définition.....	32
1.5.3 Les niveaux de participation dans une communauté de pratique	34
1.5.4 Le rôle des communautés de pratique.....	36
1.5.5 Les effets explicites des communautés de pratique.....	37
1.5.6 Les communautés de pratique virtuelles	37
CHAPITRE II	39
REVUE DE LITTÉRATURE - LOGICIELS ET LOGICIELS LIBRES.....	39
<i>2.1 Le logiciel : définition.....</i>	<i>39</i>
<i>2.2 L'économie du logiciel</i>	<i>41</i>
<i>2.3 La propriété intellectuelle dans le logiciel</i>	<i>43</i>
<i>2.4 Le logiciel libre : théories et valeurs</i>	<i>45</i>
2.4.1 Logiciel libre et logiciel open source : définition.....	45
2.4.2 Le logiciel libre dans l'économie.....	48
2.4.3 Les adversaires du libre.....	48
2.4.4 Le logiciel libre : une économie du don ?	49
2.4.5 Les structures du libre	50
2.4.6 Les incitations à la contribution dans le logiciel libre.....	51
2.4.7 Une mutation de l'univers du libre.....	51
2.4.8 Une extension de la notion de libre.....	52
<i>2.5 Les logiciels libres</i>	<i>53</i>
<i>2.6 L'innovation et la gestion des connaissances dans les logiciels libres.....</i>	<i>55</i>
CHAPITRE III.....	59
CADRE CONCEPTUEL.....	59
<i>3.1 Le logiciel libre.....</i>	<i>59</i>
<i>3.2 La gestion des connaissances</i>	<i>60</i>
3.2.1 Le modèle SECI.....	61
3.2.2 Le modèle de théorie de la firme créatrice de connaissance.....	61
<i>3.3 Notre cadre conceptuel.....</i>	<i>62</i>

CHAPITRE IV	65
MÉTHODOLOGIE	65
4.1 <i>L'approche de la recherche</i>	65
4.2 <i>Le choix méthodologique</i>	67
4.3 <i>La démarche méthodologique</i>	68
4.4 <i>Le choix du terrain</i>	71
4.5 <i>Le déroulement de la recherche et description du terrain</i>	72
CHAPITRE V	75
PRÉSENTATION DE L'UNIVERS D'UBUNTU ET DU LOGICIEL LIBRE.....	75
5.1 <i>GNU, Linux, Distributions et projets importants</i>	75
5.1.1 <i>Le système GNU/Linux</i>	76
5.1.2 <i>Les distributions</i>	76
5.1.3 <i>GNOME</i>	78
5.1.4 <i>KDE</i>	79
5.2 <i>Le projet Ubuntu</i>	81
5.2.1 <i>Description</i>	81
5.2.2 <i>Les valeurs et le fonctionnement</i>	83
5.2.3 <i>Les moyens de communication</i>	84
5.2.4 <i>Canonical</i>	86
5.2.5 <i>Les liens avec Debian</i>	86
5.2.6 <i>Le système de paquets</i>	88
5.2.7 <i>Le fonctionnement par méta-paquets</i>	91
5.2.8 <i>Quelques projets importants</i>	92
5.2.9 <i>Les distributions dérivées</i>	94
5.2.10 <i>L'écosystème d'Ubuntu</i>	94
CHAPITRE VI	96
RÉSULTATS	96
6.1 <i>L'engagement à la communauté</i>	96
6.1.1 <i>Une grande diversité des contributeurs</i>	96
6.1.2 <i>Des valeurs partagées</i>	98
6.2 <i>Un fonctionnement mérito-démocratique</i>	104
6.2.1 <i>Le droit à la parole</i>	104
6.2.2 <i>L'expertise technique</i>	115

6.2.3 Une autogestion	121
6.2.4 Des interactions décisives	128
6.3 <i>De nouvelles connaissances</i>	137
6.4 <i>De la(dis)continuité dans le temps</i>	140
CHAPITRE VII	145
ANALYSE DES RÉSULTATS	145
7.1 <i>Une description d'un modèle de création de connaissance à l'aide du modèle de la firme créatrice de connaissance</i>	145
7.2 <i>Les conditions d'activation de Nonaka présentes</i>	149
7.3 <i>Le modèle SECI et les activateurs de connaissances</i>	152
7.4 <i>Les limites de la recherche</i>	155
7.5 <i>Des perspectives d'avenir</i>	156
CONCLUSION	157
BIBLIOGRAPHIE GÉNÉRALE	159
ANNEXE 1	168
LISTE DES MAILING-LISTS UBUNTU	168
ANNEXE 2	172
CODE OF CONDUCTS UBUNTU	172
ANNEXE 3	175
"MIND MAP" DES PRINCIPALES DISTRIBUTIONS LINUX	175
ANNEXE 4	176
COPIES D'ÉCRAN DIVERSES	176
ANNEXE 5	179
SPECIFICATIONS	179
POSTFACE	187

LISTE DES FIGURES

FIGURE 1 - DISTINCTION ENTRE CONNAISSANCE ET MESSAGE D'APRÈS FORAY	16
FIGURE 2 - LES 2 PROCESSUS D'APPRENTISSAGE D'APRÈS BALLAY (BALLAY, 2002, P.35)	19
FIGURE 3 - DOUBLE BOUCLE APPRENANTE D'APRÈS ARGYRIS.....	21
FIGURE 4 - SPIRALE DE LA CONNAISSANCE D'APRÈS NONAKA ET TAKEUCHI	25
FIGURE 5 - PROCESSUS DE CRÉATION DE CONNAISSANCES ORGANISATIONNELLES (NONAKA, TAKEUCHI ET INGHAM, 1997, P.108)	27
FIGURE 6 – COMPOSANT DE LA FIRME BASÉE SUR LA CONNAISSANCE (NONAKA ET TOYAMA, 2005)	31
FIGURE 7 - NIVEAUX DE PARTICIPATION DANS UNE COMMUNAUTÉ DE PRATIQUE (D'APRÈS WENGER, MCDERMOTT ET SNYDER (WENGER, MCDERMOTT ET SNYDER, 2002).....	35
FIGURE 8 - L'ORGANISATION MATRICIELLE ET SES LOGIQUES D'ÉVALUATION, D'APRÈS BALLAY (BALLAY, 2002, P.166).....	36
FIGURE 9 - LES COMMUNAUTÉS DE PRATIQUE – LA DIMENSION DU SAVOIR, D'APRÈS BALLAY (BALLAY, 2002, P.167)	37
FIGURE 10 - FLUX DE DONNÉES, D'INFORMATION ET DE CONNAISSANCE AUTOUR DES SYSTÈMES INFORMATIQUES (D'APRÈS HORN, P.38).....	40
FIGURE 11 – RÉPARTITION DES DIFFÉRENTES CATÉGORIES DE PROJETS LIBRES, D'APRÈS FRESHMEAT.NET	53
FIGURE 13 – MÉCANISME DE CRÉATION ET TRANSFERT DE CONNAISSANCE DANS LE LOGICIEL LIBRE ...	56
FIGURE 14 - MODÈLE SECI.....	61
FIGURE 15 - MODÈLE DE LA FIRME CRÉATRICE DE CONNAISSANCE.....	62
FIGURE 16 - CADRE CONCEPTUEL	63
FIGURE 17 - CATÉGORIES ET SOUS CATÉGORIES ÉMERGENTES DE LA RECHERCHE.....	74
FIGURE 18 – ARCHITECTURE D'UNE DISTRIBUTION.....	77
FIGURE 19 - SCHÉMA SIMPLIFIÉ DE L'ARCHITECTURE MATÉRIEL / LOGICIEL.....	80
FIGURE 20 - GOUVERNANCE D'UBUNTU.....	83
FIGURE 21 - RÉCAPITULATIF DES DIFFÉRENTS TYPES ET BRANCHES DE DÉPÔTS DE PAQUETS.....	91
FIGURE 22 - ÉCOSYSTÈME D'UBUNTU	95
FIGURE 23 - EXEMPLES D'HACKERGOTCHIS DE LA COMMUNAUTÉ GNOME.....	103
FIGURE 24 - CARICATURE DU TROLL SUR INTERNET	107
FIGURE 25 - RÉCAPITULATIF DES LIENS ENTRE LOGICIELS	133
FIGURE 26 - INTÉGRATION DES LOGICIELS APRÈS LE CHANGEMENT DE LICENCE, ET LES MODIFICATIONS	136
FIGURE 27 - ILLUSTRATIONS DU TRAVAIL DANS LA COMMUNAUTÉ DU LOGICIEL LIBRE.....	137
FIGURE 28 - PRÉSENTATION D'UBUNTU COMME FIRME CRÉATRICE DE CONNAISSANCE	149
FIGURE 29 - CONDITIONS DE NONAKA VÉRIFIÉES EMPIRIQUEMENT DANS LA COMMUNAUTÉ UBUNTU	152
FIGURE 30 - MODÈLE SECI APPLIQUÉ À UBUNTU AVEC ACTIVATEUR DE CONNAISSANCES	155

LISTE DES TABLEAUX

TABLEAU 1 - TYPOLOGIE DES MONDES DE PRODUCTIONS	42
---	----

LISTE DES ABRÉVIATIONS, DES SIGNES ET DES ACRONYMES

Termes	Description	référence
Apt-get	logiciel de gestion de paquets d'une distribution basée sur Debian	ch.5
Blog	Pages web personnelles permettant généralement aux lecteurs commentaires et abonnements	ch.5
Bug	anomalie de fonctionnement d'un programme informatique	ch.2
Bugzilla	Outils de gestion de bugs	ch.2, 5
Debian	Distribution libre	ch.6
Dépôt de paquets	lieu où l'on récupère les paquets	ch.5
FLOSS	Acronyme de Free/Libre Open Source Software	ch.2
FOSS	Acronyme de Free Open Source Software	ch.2
FSF	Acronyme de Free Software Fondation, association réalisant le projet GNU.	ch.2
GNOME		ch.5
	Projet d'univers graphique englobant le gestionnaire de fenêtre et les applications	
GNU	Acronyme récursif de GNU is Not Unix, projet de création d'un système d'exploitation libre	ch.5
HIG	Acronyme de Human Interface Guide	ch.5
IRC	Espace de discussion à plusieurs sur internet	ch.5
KDE		ch.5
	Projet d'univers graphique englobant le gestionnaire de fenêtre et les applications	
KM	Acronyme de Knowledge Management (gestion des connaissances)	ch.1
LAMP	Acronyme de Linux, Apache, MySQL, Php	ch.5
Launchpad	Système de gestion sur internet développé pour par et pour les contributeurs d'Ubuntu	ch.5
Linux	Linux est le noyau du système GNU/Linux, il est toutefois souvent utilisé pour signifier le système GNU/Linux dans sa totalité	ch.5
Loco	Signifie Local Community (communauté locale)	ch.5
Mailing-list	Liste de diffusion de messages par courriels	ch.5
Mainteneur	personne chargée de créer et maintenir un paquet	ch.1
Merge		ch.5
	Phase durant laquelle les paquets de Debian sont fusionnés avec ceux d'Ubuntu	
NTIC		ch.1
	Acronyme de Nouvelles technologies de l'Information et de la Communication	
OSI	Acronyme de Open Source Initiative	ch.2
Paquet	logiciel encapsulé afin d'être compatible avec la distribution	ch.5
SABDFL	Pseudo de Mark Shuttleworth, initiateur d'Ubuntu	ch.5
Sprint	Réunion dans le monde physique de développeur Ubuntu	ch.5
SSLL	Acronyme de Société de Service en Logiciels Libres	ch.5
TIC	Acronyme de Technologies de l'Information et de la Communication	ch.2
Troll	Message "enflammé", provoquant des réactions	ch.2
Ubunteros	Utilisateur d'Ubuntu, enregistré sur Launchpad et ayant accepté et signé le code de conduite d'Ubuntu	ch.5
Unix	Type de système d'exploitation. GNU/Linux est un système de type unix	ch.5
X	Système de fenêtrage couramment utilisé sous unix	ch.5

RÉSUMÉ & ABSTRACT

Résumé

Ce mémoire présente les résultats et l'analyse d'une recherche effectuée dans la communauté Ubuntu, menée entre octobre 2005 et août 2006, dans le domaine de la gestion des connaissances, visant à comprendre comment se crée la connaissance dans la communauté, dans un contexte d'écosystème en mutation.

Il ressort de cette recherche que la grande diversité des contributeurs est maintenue sous une forme communautaire par le lien des valeurs qu'ils partagent (humanisme, bidouille, liberté, partage, autodérision, démocratisation) engendrant une vision de l'informatique.

Le fonctionnement interne s'effectue sur un mode mérito-démocratique, où tous ont droit à la parole, mais où ce sont les experts et les membres les plus engagés qui ont le plus grand rôle décisionnel. La communauté est composée de nombreuses équipes autogérées, disposant d'une grande autonomie et travaillant en interaction avec d'autres équipes de la communauté Ubuntu ou d'autres communautés du logiciel libre. Cela mène à des cycles et des métacycles de création de connaissances, permettant de choisir une trajectoire technologique puis de s'en écarter par la suite.

Tout ceci permet de créer un modèle de création de connaissance, basé sur les modèles de Nonaka et Takeuchi (1995) et Nonaka et Toyama (2005). Nous avons aussi été amenés à valider les activateurs de connaissances de Nonaka (intention, variété, chaos créatif, redondance, autonomie), et à déterminer d'autres activateurs de création de connaissances propres au logiciel libre : liberté de parole, troll, outils d'extériorisation (type wiki, bugzilla, etc.), diversité des moyens de communication et des contributeurs et action même de coder.

Mots clés

Gestion des connaissances ; création de connaissance ; Ba ; activateurs de connaissance; innovation ; logiciel libre ; logiciel à code source ouvert ; communauté ; communauté de pratique ; Ubuntu.

Abstract

This dissertation presents the results and analysis from research in the Ubuntu community, conducted between October 2005 and August 2006 in the field of knowledge management with the aim of understanding how knowledge is created in the community within the context of a changing ecosystem.

Emerging from this research is the great diversity of contributors maintained under a communal form across a common set of values (humanism, *hackerism*, freedom, division, auto-derision, democratization) forming a vision of data processing.

The internal operation is performed in a merito-democratic mode, where all may participate, but where only the most committed members and experts have the more important decision-making roles. The community is composed of numerous self-managed teams enjoying high degrees of autonomy, and which work in interaction with other team of the Ubuntu community as well as with other free software communities. All of this leads to cycles and meta-cycles of knowledge creation, allowing for a technological trajectory to be chosen and subsequently changed.

All of this permits us to create a model of knowledge creation based on the models of Nonaka and Takeuchi (1997) and Nonaka and Toyama (2005). We were also able to validate the enabling conditions of knowledge from Nonaka (intention, variety, creative chaos, redundancy, autonomy), as well as determine other enabling conditions of knowledge creation pertinent to free software: namely, freedom of speech or dialogue, troll, externalization tools (type wiki, bugzilla, etc), diversity in means of communication and contributors, and the act of coding it-self.

Keywords

Knowledge management ; knowledge creation ; Ba ; activators of knowledge ; innovation ; free software ; open source ; community ; community of practice ; Ubuntu.

RÈGLES DE NOTATION PARTICULIÈRES UTILISÉES

Les sites internet référencés dans ce mémoire, sauf mention contraire, ont tous été consultés entre les mois d'octobre 2005 et de septembre 2006.

Distinction des citations suivant leurs origines

Du à la spécificité du terrain (Internet) où nous avons travaillé, nous avons été amené pour des raisons typographiques à choisir deux types de format de paragraphes suivant l'origine des citations. Ainsi, pour des extraits longs d'articles, de revues, de livres, nous avons le modèle standard :

Citations plus longues, tirées de personnes, de livres, ... Citation tirée de personnes, de livres, ... Citation tirée de personnes, de livres, ... Citation tirée de personnes, de livres, ...

Dans le cas de citations tirées d'internet (IRC, forums de discussion, *mailing-list*, etc.), nous avons choisi pour ne pas perdre d'information, une police de caractère à chasse fixe :

Citations tirées d'IRC, de *mailing-lists*, de forum ou du web, utilisées pour la recherche. Citation tirée d'IRC ou de *mailing-list*. Citation tirée d'IRC ou de *mailing-list*. Citation tirée d'IRC ou de *mailing-list*.

Identification des répondants

Pour préserver l'anonymat des personnes, nous avons remplacé dans le chapitre « résultats » les noms par les codifications <A>, , etc. Ainsi, il est possible de suivre le fil d'une discussion où plusieurs personnes interviennent. Notons qu'il n'y a pas d'unité au cours des exemples, c'est-à-dire pour illustrer que la dénomination <A> du projet Tribu ne correspond pas à la même personne que la dénomination <A> du cas d'Hugin.

« Comme la plupart des choses utiles, les ordinateurs sont une invention américaine. Regardez un ordinateur et vous verrez que c'est entièrement un produit de l'excellence américaine.

Par exemple, cet ordinateur portable IBM que j'utilise a été conçu et construit par une entreprise américaine. Il utilise des logiciels conçus par Microsoft, une des entreprises américaines les plus productives. Mon ordinateur fait tout ce que je pourrais désirer: il peut faire mon travail, calculer mes impôts ou encore rechercher dans la bible.

Comme toutes les plus belles inventions américaines, c'est un exemple d'innovation qui rend jaloux de nombreux groupes de *hackers* européens et chinois. Ils haïssent notre technologie dominante en informatique et ne s'arrêteront pas tant qu'ils n'auront pas pris le contrôle de tous nos ordinateurs.

Je parle d'un projet nommé « Linux », quelque chose que vous n'avez peut-être jamais rencontré, mais vous le ferez sûrement un jour.

C'est un programme d'ordinateur qui a été développé initialement en Finlande avec comme but de violer les droits et brevets détenus par une compagnie américaine nommée « SCO ».

Au contraire de Windows, qui est un produit commercial mature qui est normalement inclus avec chaque ordinateur, Linux est donné gratuitement. Cela peut ne pas sembler être un problème, après tout ce n'est que réaliser peu de profit de donner un produit gratuitement.

Cela serait certainement vrai si Linux n'était pas un projet issu de l'idéologie marxiste. En effet, Linux est tellement mauvais pour nos états et beaucoup d'universités qu'un expert majeur en informatique, Steve Ballmer (de Microsoft) a décrit Linux comme un cancer.

L'industrie américaine du logiciel représente plus de 7 milliards de dollars. Introduire un produit comme Linux qui peut être copié gratuitement pourrait faire s'effondrer l'industrie tout entière. Une génération entière d'utilisateurs d'ordinateurs pourrait prendre l'habitude d'accepter du logiciel de mauvaise qualité plutôt que de payer pour des produits Américains de qualité supérieure. Si seulement le danger n'était que pour notre économie ...

De nos jours les ordinateurs contrôlent tout, des stations de télévision aux porte-avions. Nos informations cruciales et notre défense sont construites sur les technologies informatiques. Si nous introduisons ce cancer dans nos réseaux, nous ne pouvons pas savoir quel effet cela pourrait avoir sur notre infrastructure.

Imaginez si le journal télévisé était piraté parce que la station de télé a décidé de faire des économies en utilisant Linux? Imaginez si un bombardier se crashait à cause d'un logiciel écrit par un *hacker* chinois ou européen anonyme. Cela aurait autant de sens que d'inviter les français à venir prendre la maison blanche.

Et devinez quel logiciel utilise Oussama Ben Laden sur son ordinateur portable ?

Si vous pensez à Linux vous avez 100% raison. Oussama utilise Linux parce qu'il sait qu'il a été conçu pour contourner les protections des DVDs, contrevenir au DMCA, et faire la faillite d'entreprises comme Disney.

La prochaine fois que quelqu'un vous demande comment Al Qaeda paye ses mitraillettes et lance-roquettes, vous pouvez lui dire que des *hackers* étrangers font un logiciel appelé Linux qui les aide à voler les munitions aux Américains.

Le problème Linux va grandissant, et n'importe quel Américain conservateur ne peut pas l'ignorer. Heureusement Microsoft a préparé une grande campagne d'information pour aider les utilisateurs à se débarrasser de cette menace. Mais il y a quelque chose que vous pouvez faire pour aider l'Amérique à rester numéro un de l'industrie informatique:

* Si un de vos amis utilise Linux ou est tenté de l'essayer, montrez lui cet article. Expliquez-lui que Linux est une grande menace et qu'en l'utilisant il ouvre probablement son ordinateur à des pirates chinois.

* Si vous voyez une entreprise utiliser Linux, c'est sûrement qu'elle n'a pas payé pour ses logiciels. Dénoncez-la à la Business Software Alliance (BSA) qui a l'autorité légale d'inspecter les ordinateurs de n'importe quelle compagnie pour y chercher des programmes illégaux comme Linux.

Enfin, rappelez-vous d'inclure les utilisateurs de Linux dans vos prières ce soir. En tant qu'individus on ne peut pas toujours changer l'esprit des gens, mais la Bible nous apprend que Dieu peut faire se repentir n'importe quel pêcheur. »

« Linux: European threat to our computers »¹
traduit par « Totalemment/cretin(s) »² et révisé.

¹ <http://shelleytherepublican.com/2006/04/linux-european-threat-to-our-computers.html>

² <http://totalementcretin.apinc.org/blog/2006/04/27/312-linux-une-menace-europeenne-pour-nos-ordinateurs>

INTRODUCTION

« "Ubuntu" est un ancien mot africain qui signifie "humanité aux autres". Ubuntu signifie également "Je suis ce que je suis grâce à ce que nous sommes tous". La distribution Ubuntu Linux apporte l'esprit Ubuntu au monde logiciel. » Tiré de <http://ubuntu-fr.org>

En ce début de XXI^e siècle, beaucoup d'individus en occident trouvent l'époque fort ennuyeuse, l'histoire serait fini ³. À propos de ces temps parfois qualifiés d'hypermodernes, nous ne partageons pas ce point de vue, au contraire, l'évolution technique s'accélère, celle-ci ayant des répercussions, souvent importantes sur nos modes de vie. Sur le plan technologique, de nombreuses mutations sont en train de se produire, et cela a de fortes incidences sur nos conceptions de l'économie et de la gestion.

Parmi ces évolutions, nous nous intéresserons dans ce mémoire plus précisément à l'une d'entre elles, à savoir le logiciel libre. Greg Kroah-Hartman a cité durant sa présentation au Symposium Linux 2006 d'Ottawa le professeur Marietta Baba⁴. Celui-ci disait : *« Open Source development violates almost all known management theories »*.

Si nous nous accordons sur le constat que le logiciel à code source ouvert ne s'explique pas par les théories de la gestion, deux attitudes sont donc possibles : la

³ Référence à « La fin de l'histoire » de Francis Fukuyama.

⁴ *Dean of the Department of Social Science à la Michigan State University*. Source : « *Myths, Lies, and Truths about the Linux kernel* », http://www.kroah.com/log/linux/ols_2006_keynote.html

première, à l'image du Alexis de Tocqueville Institution, est de nier la réalité du développement collaboratif, de ne pas tenter de comprendre et d'analyser l'origine du succès du logiciel libre, ce qui mène l'institution à déclarer que Linus Torvalds (initiateur du noyau Linux) est un voleur de code⁵, ou que les logiciels à code source ouvert sont d'inspiration communistes⁶.

L'autre attitude consiste à adopter une démarche empirique pour comprendre le fonctionnement et non l'édicter dans un univers platonique, et ce au plus près du terrain, puisque c'est là que le logiciel est conçu, est créé. Tout ceci, quitte à éventuellement remettre en cause les théories du management, si celles-ci sont erronées.

Ce mémoire s'inscrit dans cette deuxième voie, et tente, par une recherche menée sur le terrain du logiciel libre, de comprendre comment le logiciel libre peut créer autant de connaissance.

Intérêt de la recherche

Notre recherche s'effectue à l'intersection de deux domaines importants : le logiciel libre et la gestion des connaissances. Si chacun de ces deux domaines a donné lieu à de nombreux travaux, les travaux se situant à la juxtaposition des deux sont rares. L'intérêt de notre recherche est de défricher sous l'angle de la création des connaissances le domaine des logiciels libres.

⁵ Source : « Is Torvalds really the father of Linux? », http://news.zdnet.com/2100-3513_22-5216651.html

⁶ Source : « Is open source communist? », <http://blogs.zdnet.com/open-source/index.php?p=210>

Problématique de recherche

Étant issu du monde du logiciel libre et familier à son fonctionnement internet, nous avons été intéressés par les problématiques actuelles provenant du champ de la gestion des connaissances dans ce milieu.

Tout d'abord, il nous a paru intéressant de noter que le monde du logiciel libre est actuellement en pleine mutation, car de plus en plus d'utilisateurs non-développeurs s'intéressent aux logiciels libres, et des personnes n'ayant pas la culture *hacker* rentrent dans les communautés, au péril de cette dernière (Jullien et Zimmermann, 2005). Il nous semblait donc intéressant d'étudier les réponses que pouvaient apporter les acteurs du libre à ce changement d'environnement, dans une distribution Linux reconnue pour intégrer de nombreux utilisateurs « débutants ».

Par ailleurs, si les communautés sont à la base de l'innovation dans le logiciel libre (Cohendet, Creplet et Dupouët, 2003), nous nous sommes aperçus que les conditions d'activation de ces communautés n'ont pas été étudiées.

Aussi nous proposons comme problématique de recherche d'étudier les conditions d'activation des communautés du logiciel libre engendrant de nouvelles connaissances dans un contexte d'écosystème dynamique.

Pour ce faire, nous emprunterons une démarche exploratoire, directement sur le terrain, c'est-à-dire sur Internet et là où se rencontrent les diverses communautés.

Cheminement du mémoire

Dans le premier chapitre, nous traiterons de l'économie de la connaissance, et de ses caractéristiques spécifiques. Nous traiterons aussi des natures de la propriété intellectuelle et de la connaissance, et de la gestion de la connaissance.

Dans le deuxième chapitre, nous traiterons de l'univers du logiciel, de la propriété intellectuelle spécifique au logiciel, et nous décrirons les aspects théoriques et pratiques des logiciels libres tels qu'ils sont abordés dans la littérature.

Dans le troisième chapitre, nous présenterons le cadre conceptuel dans lequel s'insère ce mémoire, à la rencontre des deux champs précédemment abordés, à savoir la gestion des connaissances et le logiciel libre.

Dans le quatrième chapitre, nous traiterons de l'aspect méthodologique de la recherche. Nous définirons l'approche de la recherche suivie, les choix et démarches méthodologiques effectués, la description du choix du terrain ainsi que la description de la recherche du terrain.

Dans le cinquième chapitre, nous présenterons au lecteur l'univers d'Ubuntu ou la recherche a été effectuée, ainsi que plus généralement l'univers du logiciel libre.

Dans le sixième chapitre, nous présenterons les résultats de la recherche exposant la nature de l'engagement à la communauté, son mode de fonctionnement, ses types de création et son mode de production.

Enfin, dans le septième chapitre, nous analyserons les résultats obtenus en les confrontant au cadre conceptuel présenté au chapitre 3, construit à partir des chapitres 1 et 2, et en tirerons des enseignements sur la création de la connaissance dans le domaine du logiciel libre.

CHAPITRE I

REVUE DE LITTÉRATURE – GESTION DE LA CONNAISSANCE

Dans ce chapitre, nous traiterons de l'économie de la connaissance, qui avec l'émergence des technologies de l'information et de la communication est le contexte dans lequel ce mémoire s'insère. Puis nous traiterons de la nature de la connaissance, notamment dans l'univers de la gestion, ce qui nous mènera naturellement à la gestion de la connaissance. Nous aborderons alors les concepts de « Ba » et de communautés de pratiques, qui sont une des pratiques possibles facilitant les transferts et création de connaissance.

1.1 L'économie de la connaissance

1.1.1 Les TIC et nouvelle économie ?

L'économie de la connaissance, aussi connue sous le nom « d'économie du savoir » ou pendant la bulle internet, sous le vocable de « nouvelle économie », est un phénomène assez récent dans l'histoire économique.

En fait, si ces termes peuvent signifier la même chose, il y a toutefois des nuances à effectuer. Le terme de « *nouvelle économie* » provient de l'hebdomadaire News Week (Mandel, 1996) et reflète pour Blondeau (2000c) une vision proche des courants financiers, dont le but est de « *gérer, sinon conjurer les effets profondément déstabilisateurs de la crise de rentabilité du capital, s'exprimant depuis près de vingt ans dans la spirale financiariste* ». Pour l'auteur, cette expression dénote une tendance à la marchandisation du savoir, ce qui représente « *le défi auquel sont*

confrontés les tenants de l'ordre néo-libéral ». Pour l'auteur, reprenant Castells⁷, la manière de concilier industrialisme (s'inscrivant dans une logique de croissance économique) et informationnalisme (s'inscrivant dans une logique de développement technologique et d'accumulation des savoirs) passe non pas par la nouvelle économie, « *version techno-remixée de notre bon vieux capitalisme fordien* », mais par une économie du savoir.

Pour juger de l'importance des NTIC (Nouvelles Technologies de l'Information et de la Communication), en 2003 au Canada, « *7,9 millions (64 %) des 12,3 millions de ménages canadiens comptaient au moins un membre utilisant régulièrement Internet en 2003* » (Statistique Canada, 2004). Et entre 2003 et 2005, « *la proportion de ménages ayant déclaré un accès à haute vitesse étant passée de 35 % en 2003 à 43 %* » (Statistique Canada, 2005).

Toutefois, l'ampleur de l'économie de la connaissance ne fait pas l'objet d'un consensus chez les économistes. Ainsi, si Stoffaës (2000, p. 254) met en parallèle les nouveaux réseaux avec les chemins de fer au XIX^e siècle et avec l'électromania au début du XX^e siècle, pour Didier (2000, p. 126), « *l'histoire économique est depuis longtemps un enchaînement permanent de nouvelles économies* ». Didier met en parallèle Internet et les NTIC avec les innovations du XIX^e et XX^e siècle. Il compare la progression du téléphone portable avec celle de la radio au début du siècle dernier, et précise que l'ordinateur personnel s'est répandu moins vite que la télévision couleur. Didier se demande si la croissance des nouveaux secteurs s'ajoute à celle des secteurs traditionnels, ou si elle s'opère au moins en partie par substitution de nouveaux services à d'autres. Selon Gadrey (2000, p. 20), la nouvelle économie relève plus du mythe que du modèle, car elle subit une épuration au travers de laquelle seules certaines caractéristiques idéalisées sont retenues. Ces caractéristiques

⁷ Castells, Manuel (1998), « la société en réseau », Fayard.

de la nouvelle économie sont essentiellement transmises par « *la presse économique et d'affaires, commentateurs de la bourse, personnalités politiques s'appuyant sur la presse économique, et réciproquement* ». Gadrey oppose à la nouvelle économie ce qu'il appelle l'économie (parfois) nouvelle. Foray (2000, p. 18) note que si « *la connaissance a toujours été au cœur du développement économique* », la principale différence réside ici dans la forte augmentation de la proportion de capital intangible, et l'irruption des TIC.

Les NTIC constituent pour Plihon (2003) l'essence d'une troisième révolution industrielle. En effet, l'auteur explique que les NTIC ont un effet sur l'ensemble de la société, ainsi que sur les secteurs traditionnels de l'économie, via, ce que Hervé Sérieyx appelle un double effet de cannibalisation (destruction de pans entiers d'activités) et de pollinisation (dynamisation des entreprises).

Les NTIC permettent d'accroître encore l'économie de service, voire d'effacer la frontière entre biens et services. C'est l'entrée dans « *l'économie de l'immatériel* ». Rifkin⁸ développe la thèse de « *l'économie de l'accès* » où l'accès remplace la propriété, la location remplace l'achat. Les objets deviennent ainsi mouvants. Pour Sterling (2000), « *ce n'est même pas, en vérité, l'accès lui-même, mais les indications qui disent à quoi il faut accéder, - à quoi il faut prêter attention* ».

Pour Pierre Veltz cité par Plihon, grâce aux NTIC, le nouvel objectif de l'entreprise devient la recherche de flexibilité. Pour Reich, on passe ainsi au concept « *d'entreprises-réseaux* ». Les NTIC permettent la transversalité de l'information, permettant ainsi de nouvelles structures managériales. L'organisation conçue comme statique devient ainsi dynamique. Dans ce nouveau paradigme, les sources d'efficacité sont la créativité technique et le savoir-faire commercial.

⁸ Rifkin, Jeremy (2002), « *L'âge de l'accès : la révolution de la nouvelle économie* », Paris, La découverte.

Au sein de cette nouvelle économie, on assiste à une importante montée en puissance de la finance qui est fortement liée aux NTIC pour Plihon, car la finance est fondée sur le traitement de l'information. Aussi les NTIC ont permis d'accélérer et de renforcer le rôle du système financier, et celui-ci a fortement contribué au développement des NTIC.

Les NTIC ont pour effet un accroissement de la productivité du capital car le capital informatique permet de produire plus à des coûts de plus en plus bas. Néanmoins, Robert Solow note un paradoxe : le gain de productivité des NTIC est très fort dans le secteur des NTIC, mais dans les secteurs traditionnels de l'économie, il est beaucoup moins important. Plihon avance trois hypothèses explicatives : le gain de productivité des NTIC est d'ordre qualitatif et serait sous-estimé; les NTIC n'offrent un gain de productivité que dans les entreprises qui ont su s'organiser autour; la diffusion des NTIC n'est pas un phénomène immédiat en raison de l'inertie du au changement de technologie.

Jacquet voit l'ouverture de marché et la concentration de la recherche comme un formidable incitatif à l'innovation :

À la réflexion cependant, ce n'est pas l'abondance de l'information qui compte, mais la qualité et le traitement de cette information. Paradoxalement, en facilitant la circulation de l'information, les TIC peuvent avoir pour conséquence d'en détériorer la qualité, de faciliter la désinformation (Jacquet, 2000, p. 74).

1.1.2 Le savoir dans l'économie

D'après le rapport de l'OCDE (2000, p. 13), les modèles économiques voient la connaissance comme le moyen de recueillir et de traiter l'information requise pour faire des choix, ainsi que comme un actif contribuant à la production grâce à la compétence et l'innovation.

Pour Artus (2000, p. 39), une telle économie produit à la fois de nouveaux bienfaits et de nouveaux problèmes. Elle nécessite une accumulation du capital et une main-d'œuvre qualifiée, mais soulève des enjeux sociaux (augmentation et accroissement des inégalités) et financiers (valorisation des entreprises). Pour l'auteur, le capital s'y substitue au rôle traditionnel du travail, ce qui pose une nouvelle forme de conflit de répartition entre la rentabilité du capital et la rémunération du travail. Artus s'interroge sur la productivité réelle de cette économie :

est-on certain que l'économie sur la masse salariale avec les gains de productivité n'est pas plus qu'annulée par la hausse des coûts associés à la hausse de l'intensité capitaliste.

D'après le rapport de l'OCDE (2000, p. 19), aussi bien l'économie classique qui voit l'accès à la connaissance comme universel, qu'à l'opposé, la théorie de l'entreprise qui voit la connaissance comme uniquement accessible aux entreprises font erreur. Pour les auteurs du rapport, le savoir se situe entre les deux extrêmes.

D'après Foray (2000, p. 59), contrairement aux conceptions classiques considérant la connaissance comme allant de soi, l'économie de la connaissance s'intéresse à la création et la diffusion du savoir. D'autant plus que la connaissance possède trois caractéristiques distinctives qui la différencient des objets économiques classiques : elle est non excluable (difficulté de contrôle du bien), non rivale (peut appartenir à plusieurs personnes) et cumulative (la connaissance engendre de nouvelles connaissances). Pour Plihon (2003), dans une logique d'économie de la connaissance, on ne peut se référer à l'idée économique selon laquelle le prix d'un produit doit tendre vers son coût marginal, puisque le coût marginal tend vers 0 (ou presque). Par ailleurs, les NTIC remettent aussi en question d'autres propriétés fondamentales de l'économie tels la transparence, le caractère exclusif de la consommation d'un bien et son caractère rival. Dans ce nouveau contexte, *« on se trouve en présence de monopoles naturels et de biens publics »*. Pour pallier cela dans notre contexte concurrentiel, les entreprises se différencient.

Toujours d'après Foray (p. 39), d'un point de vue économique classique, la connaissance se trouve dans les départements de R&D et dans les universités. Il s'agit d'une production de connaissance délibérée. Mais pour l'auteur, au-delà de la vision économique traditionnelle, la connaissance se trouve aussi dans la pratique d'un processus, relevant d'une production de connaissance non délibérée (« *learning by doing* » ou « *learning by using* »).

Ainsi, Foray conclut que le problème de l'économie de la connaissance est un problème de reproduction devant tenir compte de la capacité d'apprentissage, là où l'économie de l'information vise plutôt à protéger l'information.

1.1.3 L'importance de l'innovation

Dès lors, dans l'économie du savoir il devient important d'innover. Le premier à l'avoir analysé est Schumpeter (Heilbroner, 1977), qui analyse l'économie capitaliste, et la décompose en une composante statique et une composante dynamique. Pour Schumpeter, le profit vient uniquement de la composante dynamique, c'est-à-dire l'innovation créée par l'élite de la société capitaliste, les entrepreneurs. Ainsi, l'entrepreneur innove au mépris des risques, et lorsqu'il réussit, il crée du profit sur une courte période avant d'être rejoint par la concurrence qui n'a plus qu'à suivre le chemin. Pour Schumpeter, l'entrepreneur risque d'avoir le rôle du capitaliste chez Ricardo : même si c'est lui qui apporte le profit, ce n'est pas lui qui en profite.

Pour Schumpeter, les monopoles sont à la base d'un cycle qu'il nomme de destruction créatrice. L'entreprise monopolistique est l'entreprise la plus innovante parmi un ensemble d'entreprises où chacune se bat pour occuper cette place dominante. Mais une fois en position de monopole, l'entreprise perd sa capacité à innover et à provoquer du changement, laissant ainsi la chance à de nouvelles entreprises plus petites d'innover et de détrôner l'entreprise détenant le monopole.

Pour Barreyre (Ebrahimi et Saives, 2006, p. 463), il y a trois types d'innovation : les innovations à connotation technologique touchant au produit ou service, les innovations à connotation commerciale portant sur les fonctions de distribution et de marketing, et les innovations à connotation organisationnelle portant sur le processus d'administration et de développement.

Pour Kline et Rosenberg (1986), le processus de l'innovation n'est pas un processus linéaire, mais au contraire est un processus dynamique basé sur l'interaction entre les différents acteurs, des producteurs aux utilisateurs.

Pour Foray (2000, p. 75), il y a deux systèmes d'incitation et de coordination qui règlent les problèmes d'externalités : créer un marché, qui est dès lors régulé par des droits de propriété (droit d'auteur, brevet, ...); créer au sein du système public (université, laboratoires publics, ...), ce qui ne nécessite pas de droits exclusifs.

D'après le rapport de l'OCDE (2000, p. 23), l'innovation est une résultante intéressante de la création de savoirs pour deux raisons :

- l'innovation correspond à une création et augmente le savoir existant;
- l'innovation est un savoir qui fait l'objet d'une demande.

Aussi, l'innovation est un passage important de l'arrivée de nouveaux savoirs dans une économie capitaliste.

Pour Plihon (2003), les NTIC accélèrent l'innovation, en contrepartie du raccourcissement des durées de vies. « *Avec les NTIC, la variable essentielle n'est plus l'espace en voie d'être maîtrisée mais le temps* ».

1.1.4 La protection de la propriété intellectuelle

Plusieurs auteurs s'interrogent sur la nature de la propriété intellectuelle. S'agit-il de « droits naturels » ? Latrive (2004) étudie la question en mettant en opposition les

arguments des deux camps : pour Bastiat, « *nous naissons tous propriétaires* ». L'œuvre intellectuelle, pour Napoléon « *est une propriété comme une terre, comme une maison; elle doit jouir des mêmes droits* ». À l'opposée, Jefferson compare la circulation des idées avec la flamme d'une bougie : éclairer son prochain ne requière nul sacrifice, il est donc antinaturel de vouloir protéger la propriété. Pour Victor Hugo :

Le livre, comme livre, appartient à l'auteur, mais comme pensée, il appartient – le mot n'est pas trop vaste – au genre humain. Toutes les intelligences y ont droit. Si l'un des deux droits, le droit de l'écrivain et le droit de l'esprit humain, devait être sacrifié, ce serait, certes, le droit de l'écrivain, car l'intérêt public est notre préoccupation unique, et tous, je le déclare, doivent passer avant nous.

Samudrala (2000) remarque que la loi fédérale américaine⁹ dément le droit naturel :

La mise en œuvre par le Congrès d'une législation sur le droit de reproduction conformément à la Constitution, ne repose pas sur un droit naturel que l'auteur aurait sur ses œuvres [...] mais sur l'idée que le bien public et le progrès de la science et des arts utiles seront favorisés.

Thomas Jefferson (Quéau, 2000) écrivait: "Les inventions ne peuvent pas, par nature, être sujettes à la propriété."

Ainsi, pour Samudrala (2000), « *le "progrès de la science et des arts utiles" constitue une raison éthique valable de restreindre la liberté individuelle de reproduire, modifier et diffuser des informations déjà disponibles, et [...] il est louable de consacrer des ressources sociales à restreindre ces libertés en vue du bien public* ». Mais l'auteur émet des doutes quant au bien-fondé de l'amélioration du progrès de la science et des arts dans tous les cas de figure, notamment dans le domaine de l'industrie musicale et du logiciel.

Pour Liotard (1999), les droits de propriété industrielle ont à l'origine pour objectif de concilier les intérêts privés de l'innovateur et les intérêts de la société. Les droits de

⁹ Copyright Act, 1909

propriété intellectuelle peuvent être séparés entre outils de propriété industrielle (brevets, plans) et outils de propriété littéraire et artistique (droits d'auteurs). Ils ont une fonction de protection de l'invention : ils accordent des droits sur l'usage et l'exploitation à l'innovateur. Pour être accordé à une invention, un brevet doit respecter trois conditions : la nouveauté, l'inventivité (c'est-à-dire la non-évidence pour l'homme de métier) et l'application industrielle. Un brevet comporte une requête, une description de l'invention et des revendications. Chabchoub distingue les caractéristiques des brevets européens ou les brevets doivent avoir un caractère technique des brevets américains qui doivent être utiles, l'utilité étant moins contraignante pour le détenteur du brevet à prouver que le caractère technique (Chabchoub, 2005, p. 76).

En contrepartie, l'invention devient publique au bout d'un certain temps et l'ensemble de la société dispose alors du savoir. Ainsi, la durée et l'étendue du brevet sont à balancer pour trouver l'équilibre entre droits de l'inventeur et de la société.

L'idée d'une entreprise dont les actifs seraient immatériels, car basés sur les connaissances et le savoir remettrait ainsi en cause la notion d'économie classique de valeur-travail d'après Plihon (2003). Les connaissances, de nature publique, seraient en contradiction avec le fait que d'elles dépendent sources de valeur et de profits des entreprises. Celles-ci essaieraient de limiter cette contradiction en privatisant les connaissances (via le droit de propriété intellectuelle).

Toujours selon Liotard, les brevets s'appuient les uns sur les autres pour innover, et servent ainsi d'instrument de transfert de connaissances, permettant de faire progresser les savoirs. Mais pour l'auteur, pour vraiment maîtriser le brevet, il est nécessaire de passer par des accords de licences.

Ainsi, la licence d'un brevet peut servir à une entreprise à augmenter ses gains, à garder son avance technologique ou encore à garder son rôle de *leader* sur le marché.

Néanmoins dans le jeu de la concurrence, le brevet qu'une entreprise dépose lui permet de donner des informations à ses adversaires économiques : cela leur permet depuis l'extérieur d'analyser les recherches d'une entreprise, ou encore de donner un chemin à suivre.

Toutefois, Liotard note que plus récemment, le brevet a été détourné de son usage premier pour un usage stratégique que peut en faire l'entreprise, ce qui n'avait pas été pensé lors de la mise au point des mécanismes de propriété intellectuelle. La firme se sert du brevet pour conquérir de nouveaux marchés, conserver et défendre ses positions acquises face à ses concurrents potentiellement dangereux, barrer le passage à des firmes rivales, piéger les concurrents en publiant des brevets visant à les embrouiller, interdire l'exploitation à des tiers.

Mais ce nouvel usage ne fait pas l'unanimité. Pour certains auteurs, il est néfaste pour la société et pour l'innovation. Ainsi, selon Cohen, « *En l'espace de vingt ans, on est en fait passé de la présomption que le brevet devait être l'exception [...] à la situation inverse où la propriété intellectuelle est devenue la règle. Cette idée est dangereuse* » (Cohen, 2003, p. 174). Henry et al. prennent l'exemple de la création d'un « champ de mines », c'est-à-dire d'un ensemble de dépôt de brevet visant à gêner la concurrence. Il peut être assez simple de déposer des brevets sur des méthodes utilisées par des firmes concurrentes, même si les brevets ne sont pas nécessairement valides. Celle-ci doit alors tenter de faire annuler le brevet devant un court de justice, ce qui en plus d'être aléatoire diminue l'efficacité globale de la société en ajoutant des externalités négatives. De plus, le système judiciaire ne pourrait faire face longtemps à une telle augmentation de procédures, et serait rapidement engorgé (Henry, Trometter et Tubianna, 2003, p. 104).

Si la protection intellectuelle n'est pas forcément corrélée à l'innovation comme le pense Berlan (2003), pour qui aucune étude scientifique ne prouve que les droits de

propriété intellectuelle favorisent l'innovation, selon l'avis de tous, son absence engendrerait un recul de cette dernière. Ainsi, pour beaucoup, le système de protection intellectuelle devrait être revu. Selon Henry et al.,

le système de protection de la propriété intellectuelle par des brevets tel qu'il fonctionne actuellement dans les pays développés est malade, singulièrement aux États-Unis. La maladie est plus manifeste et plus aiguë dans le secteur biomédical, mais elle n'y est pas confinée. Et c'est ce système malade qu'on s'est mis en devoir d'imposer au reste de la planète dans le cadre de l'OMC, par les accords ADPIC (TRIPS) (Henry, Trometter et Tubianna, 2003, p. 99).

Les droits de propriété intellectuelle concernant le logiciel étant atypiques et importants dans le cadre de ce mémoire, ils font l'objet d'une partie à part dans le chapitre suivant à la section 2.3.

1.1.5 Une nouvelle forme d'organisation : l'organisation en réseau

Pesqueux et Ferrary (Pesqueux, 2002, p. 198, Ferrary et Pesqueux, 2004, p. 19) décrivent l'organisation en réseau comme le mode d'organisation de la nouvelle économie permettant

un recentrage sur les activités, l'éclatement des centres de décision et la circulation 'transversale' des informations allant de pair avec un affaiblissement supposé de la hiérarchie, même si on reconnaît qu'elle puisse ne pas disparaître complètement.

Ferrary et Pesqueux (2004, p. 31) dénombrent quatre champs disciplinaires s'intéressant aux réseaux : l'ingénierie, l'économie, la gestion et la sociologie. Pour les auteurs, le modèle du réseau vient remplacer le modèle du système, les deux étant basés sur la cybernétique, mais le modèle en réseau ayant comme avantage sur le modèle du système d'être dynamique (un système une fois défini est statique, alors que les acteurs d'un réseau peuvent sans cesse changer). Cette nouvelle forme aboutirait ainsi au « *nec plus ultra* de l'innovation » et à l'effacement des imperfections vers l'idéologie du marché pur et parfait. Ainsi, le concept de réseau repose sur l'idée de changement permanent, d'espace-temps toujours remodelé et

d'innovation organisationnelle permanente (Pesqueux, 2002, p. 209). Le réseau permet de passer d'une coordination par le contrôle à une régulation par la coopération (p. 220). Néanmoins, malgré une apparence de moindre pouvoir, le réseau promeut une nouvelle forme de pouvoir d'après Pesqueux, s'appuyant sur l'analyse de Foucault¹⁰.

1.2 La connaissance

Avant de nous intéresser à la gestion des connaissances proprement dites, il semble nécessaire de faire détour autour de la connaissance elle-même. Étant donné la grande quantité de disciplines s'intéressant à la connaissance (philosophie, psychologie, éducation, ...), nous traiterons cette partie à partir d'auteurs en gestion.

Pour Foray (2000, p. 9),

La connaissance possède quelque chose de plus que l'information; elle renvoie à la capacité que donne la connaissance à engendrer, extrapoler et inférer de nouvelles connaissances et informations.

La connaissance est à distinguer du message contenant cette connaissance tel qu'on peut le voir sur la Figure 1. Ainsi, la connaissance est forcément liée à un individu.

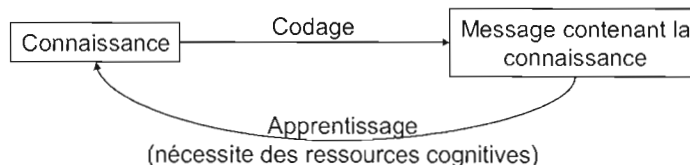


Figure 1 - Distinction entre connaissance et message d'après Foray

Nonaka et Takeuchi (1997, p. 76) considèrent que l'information comme la connaissance ont toutes les deux rapport à une signification. Toutefois, les auteurs distinguent la connaissance de l'information sur deux points :

¹⁰ Foucault, Michel, « Surveiller et punir ».

- la connaissance requière une croyance, une adhésion;
- la connaissance concerne l'action.

Nonaka et Takeuchi voient l'information comme étant « *un flux de messages alors que la connaissance est créée par ce flux d'information et est ancrée dans les croyances de celui qui la détient* » (1997, pp. 77-78).

Prax (2000) décrit les quatre formes de connaissances selon les Grecs :

- l'*épistémè*, connaissance abstraite généralisante;
- la *techné*, connaissance permettant l'accomplissement d'une tâche;
- la *phronesis*, sagesse sociale;
- la *mètis*, connaissance conjecturale, ruse, flair.

Pour Lundvall et Johnson, cités dans le rapport de l'OCDE (2000, p. 15), il existe quatre formes de savoir :

- le savoir factuel : désigne la connaissance des faits;
- le savoir intellectuel : désigne la connaissance des principes et des lois qui régissent le mouvement dans la nature;
- le savoir-faire : désigne les compétences, la capacité à faire quelque chose;
- le savoir relationnel : désigne la capacité sociale à coopérer et à communiquer avec différentes personnalités et différents experts.

Gibbons (1994) propose une distinction entre deux modes de production du savoir : il désigne par « mode 1 » un savoir de type scientifique et évalué par les pairs, alors que par « mode 2 », l'auteur entend un savoir plus appliqué, centré sur les problèmes, transdisciplinaire et hétérogène. Si le mode 1 a pour moteur la créativité individuelle, le « mode 2 » repose quant à lui sur la créativité collective. Néanmoins, c'est le « mode 1 » qui est traditionnellement dominant et mieux reconnu.

Nonaka et Takeuchi (1997, ch. II) voient deux grands courants traitant la connaissance sous un angle complètement différent : la tradition occidentale et la tradition orientale. Pour l'épistémologie dans la tradition occidentale, deux tendances s'opposent entre rationaliste (Platon, Descartes, Kant, ...) et empiriste (Aristote, Lock, Marx, ...). Mais dans les deux cas, seule la « *connaissance vraie, justifiée* », objective est étudiée. Dans la tradition orientale, la connaissance est vue à travers la nature, le corps, l'être. Elle y est comprise sous un aspect beaucoup plus subjectif. Ainsi, si en occident, c'est l'aspect croyance vraie qui est mis de l'avant, les auteurs entendent quant à eux s'intéresser davantage à la croyance justifiée (Nonaka et Takeuchi, 1997, p. 78).

Pour Ballay (2002, p. 49), la connaissance est à la fois objective et subjective. Elle est liée au corps, et relève à la fois de la mémoire et de la pensée. L'auteur (p. 23) voit quatre modalités de la connaissance :

- la pensée rationnelle : elle produit et utilise le savoir;
- la pensée sensible : elle produit et utilise l'intuition;
- la mémoire procédurale : elle code le savoir-faire;
- la mémoire épisodique : elle code l'information.

Ballay (p. 35) voit aussi deux types d'apprentissage : l'apprentissage en acte (liée à l'action) et l'apprentissage rationnel (liée au concept). Les deux types d'apprentissages sont complémentaires et permettent de passer de l'information au savoir et à la compétence, tel que schématisé sur la Figure 2.

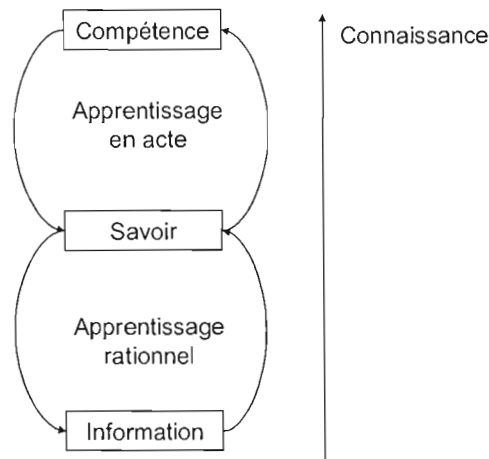


Figure 2 - Les deux processus d'apprentissage d'après Ballay (2002, p. 35)

Ballay (p. 50) voit deux fonctions à la connaissance :

1. prescrire : ce processus menant à l'action;
2. interpréter : il s'agit ici de donner du sens.

Polanyi (1966, p. 4) divise la connaissance en deux catégories, explicite et tacite. La connaissance explicite correspond à la connaissance mesurable, exprimable aisément en mot ou en chiffre, et donc plus propice à la codification. Par opposition, la connaissance tacite est ce qui est connu mais difficilement exprimable. C'est ce qui fait dire à Polanyi que « *nous savons plus que nous pouvons exprimer* ». Il n'est ainsi pas nécessaire de savoir expliquer le fait de tenir en équilibre sur une jambe par des matrices inertielles pour savoir le faire. Cette catégorisation semble aujourd'hui faire l'unanimité.

Pour Nonaka et Takeuchi, on peut parler de connaissance collective, en ce sens que même si la connaissance est toujours individuelle, elle peut être supportée par l'organisation (1997, p. 79). C'est la dimension ontologique de leur modèle. Ainsi,

Creating knowledge *organizationally* [...] means that subjective tacit knowledge held by an individual is externalized into objective explicit knowledge to be shared and synthesized. [...] Knowledge is socially created through the

synthesis of the different views held by various people. (Nonaka et Toyama, 2005, p. 422).

1.2.1 L'apprentissage

D'après le rapport de l'OCDE, l'économie classique comprend le processus d'apprentissage comme étant soit assimilé à de « l'acquisition d'informations », soit passant par une augmentation de la productivité. Cette représentation de l'apprentissage est dépassée. Pour les auteurs,

l'apprentissage se définit comme un processus au cœur duquel on trouve l'acquisition de compétences et de qualifications permettant à l'individu apprenant de mieux atteindre ses objectifs individuels ou bien ceux de son organisation. (OCDE, 2000, p. 32).

Pour Foray, il existe deux niveaux d'apprentissage : l'apprentissage de nature routinière et l'apprentissage par l'expérience. À travers l'expérience, la production de connaissance peut être collectivement distribuée. Mais il devient alors nécessaire d'apprendre à utiliser cette connaissance, à la valoriser et la capter. C'est, pour l'auteur, ce que fait l'entreprise apprenante (Foray, 2000, p. 39).

1.3 La gestion de la connaissance

1.3.1 Les origines

Argyris et Schön voient dans la notion de double boucle d'apprentissage l'élément déterminant de l'entreprise apprenante (Argyris et Schön, 1978). Là où une première boucle modifie les actions conduisant à des conséquences non désirées, la double boucle modifie les valeurs à la base de ces actions (voir Figure 3).

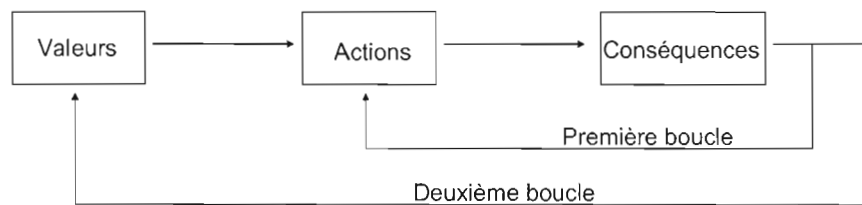


Figure 3 - Double boucle apprenante d'après Argyris

Selon Senge (1991, pp. 21-28), l'organisation intelligente n'est possible qu'à travers un type de pensée systémique. C'est ce qu'il appelle la cinquième discipline, pierre angulaire des quatre autres : maîtrise personnelle, remise en question des modèles mentaux, vision partagée et apprentissage en équipe.

Trois théories sont à l'origine de l'émergence de la gestion des connaissances dans le domaine managérial (Ebrahimi et Saives, 2006, p. 456). Tout d'abord, la théorie des ressources (ou *resource based view*, RBV) où l'entreprise se distingue de la concurrence grâce à ses compétences. Puis la théorie des capacités dynamiques (*dynamic capabilities*) décrit l'entreprise qui réussit en sachant mêler les bonnes compétences (les savoirs fondamentaux) aux bonnes capacités (savoir-faire), tout en sachant se renouveler. Enfin, la théorie des compétences clés (*core competencies*) s'intéresse à la valorisation et l'exploitation des compétences dans l'entreprise : il s'agit de faire correspondre les compétences et capacités aux nécessités de l'entreprise.

1.3.2 La gestion des connaissances

Foray oppose deux visions pour l'entreprise afin de gérer la connaissance : d'un côté, rendre toute forme de connaissance explicite à travers une codification systématique des données, d'un autre côté, accepter que la connaissance reste tacite et faciliter son transfert dans l'organisation entre individus (Foray, 2000, ch. IV).

Lundvall fait la distinction entre le modèle oriental, et le modèle occidental de création de connaissance (2000, p. 154). Le modèle occidental est hiérarchisé, la création s'effectuant en haut et les niveaux inférieurs exécutant, et est beaucoup basé sur la concurrence. Au contraire, dans le modèle oriental, la création a lieu dans les niveaux intermédiaires (Nonaka et Takeuchi, 1997, p. 148) et s'établit sur des relations moins concurrentielles et plus basées sur le long terme.

Nous plaçons dans une première école de pensée, l'école technologique, ce que Foray qualifie de codification systématique et Lundvall de modèle occidental. Nous rapprochons l'école organisationnelle de l'acceptation de l'existence de la connaissance tacite chez Foray et du modèle oriental chez Lundvall.

1.3.2.1 L'école technologique

Dans cette approche, la gestion de connaissance se doit de rendre codifiable et diffusable le maximum de savoir, de transformer le plus possible de connaissance tacite en connaissance explicite, et de la stocker dans des bases de données.

Prax (2000) tente de définir la gestion de la connaissance ainsi :

- Le KM est une approche qui tente de manager des items aussi divers que pensées, idées, intuitions, pratiques, expériences émis par des gens dans l'exercice de leurs professions
- Le KM est un processus de création d'enrichissement, de capitalisation et de diffusion des savoirs qui implique tous les acteurs de l'organisation, et tant que consommateurs et producteurs.
- Le KM suppose que la connaissance soit capturée là où elle est créée, partagée par les hommes et finalement appliquée à un processus de l'entreprise.

Pour Davenport et Marchand (2000), la distinction primordiale entre gestion de l'information et gestion des connaissances est que cette dernière s'intéresse en plus à la création des connaissances, et à leur utilisation.

Pour Tisseyre (1999, pp. 22-23), le *knowledge management* relève de trois adjectifs importants : la prise de conscience, la coordination, le résultat opérationnel attendu. La gestion de la connaissance se fait à travers quantité d'outils dans le but de créer de la valeur, se mesurant à travers des outils tel le ROI (*Return On Invest*) (p. 85).

Pour Ballay (2002, p. 86), l'objet de la gestion des connaissances est de quatre ordres :

- la capitalisation des connaissances : l'ensemble des processus de récupération des connaissances ayant pour rôle d'évaluer le seuil de valorisation;
- le transfert des connaissances : l'ensemble des processus permettant à des individus n'ayant pas la connaissance qu'ils l'acquièrent, le transfert ayant pour rôle d'augmenter la valeur d'usage;
- le renouvellement des connaissances : l'ensemble des processus améliorant l'état des connaissances ayant pour rôle de régénérer la richesse;
- la socialisation des connaissances : l'ensemble des processus sous tendant les relations entre individus, à la base des trois autres processus, ayant pour rôle d'accélérer le processus de valorisation.

Bien que Ballay reconnaisse l'importance de la socialisation, celle-ci n'est présente qu'en tant que « *facilitateur* » pour les trois autres processus.

1.3.2.2 L'école organisationnelle

Les tenants de cette approche considèrent la gestion des connaissances comme un processus avant tout humain. La technologie peut y jouer un rôle, mais elle n'est qu'un support à la création. De plus, le rôle de la connaissance tacite est considéré comme étant bien plus important que dans l'approche technologique. Ainsi, Nonaka et Takeuchi estiment que « *la connaissance est profondément enracinée dans l'action et l'expérience d'un individu autant que dans ses idéaux, ses valeurs et ses émotions* » (1997, p. 26). Ils précisent plus loin que « *les impressions, intuitions et*

soupçons hautement subjectifs font partie intégrante de la connaissance. La connaissance couvre aussi les idéaux, les valeurs et émotions autant que les images et les symboles » (p. 27). Le postulat fondamental dont partent les auteurs dans leur modèle est que « la connaissance humaine est créée et étendue au travers de l'interaction sociale entre connaissances tacites et explicites » (p. 81).

Nonaka et Takeuchi décrivent la création de connaissances collectives comme une spirale passant par quatre étapes (pp. 83-93) :

- la socialisation (tacite / tacite) : processus de partage d'expériences créant de la connaissance tacite;
- l'extériorisation (tacite / explicite) : processus d'articulation de connaissance tacite en concepts explicites sous forme de métaphores, d'analogies, d'hypothèses ou de modèles;
- la combinaison (explicite / explicite) : processus de systémisation de concepts en un système de connaissance, qui combine différents corps de connaissances explicites. La combinaison s'effectue à travers des médias tels les réunions, conversations, ...
- l'intériorisation (explicite / tacite) : processus d'incorporation de la connaissance explicite en connaissance tacite. L'intériorisation est étroitement liée à l'apprentissage en faisant.

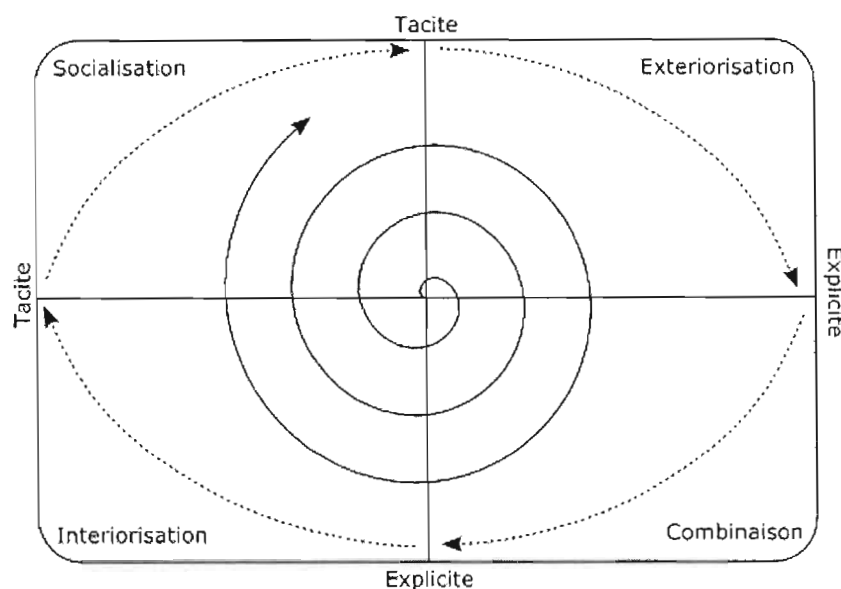


Figure 4 - Spirale de la connaissance d'après Nonaka et Takeuchi

Ebrahimi et Saives, en se basant sur le modèle de Nonaka et Takeuchi, énoncent quatre étapes correspondant aux quatre phases : la socialisation nécessite un espace de partage, l'extériorisation nécessite un artefact, la combinaison s'appuie sur la redondance et l'intériorisation sur l'appropriation (Ebrahimi et Saives, 2006, pp. 482-483).

Dans ce mémoire, nous nous référerons à la définition d'Ebrahimi et Saives qui définissent la gestion des connaissances comme étant

[...] un processus d'apprentissage, de création, de transformation, de circulation des savoirs explicites et tacites dans un contexte donné, effectué par les hommes, intégrés dans les différents processus de l'organisation, soumis à la logique managériale en vigueur. Elle est cumulative et non rivale (, p. 460).

1.3.3 Les conditions et phases de la création de connaissance

Pour Nonaka et Takeuchi (1997, pp. 96-107) il est nécessaire de réunir cinq conditions afin de favoriser la création de connaissance dans l'entreprise :

- l'intention : c'est l'aspiration de l'organisation vers ses buts;

- l'autonomie : les individus doivent être autorisés à agir de façon autonome, pour autant que les circonstances le permettent;
- la fluctuation et le chaos créatif : permettent une rupture des routines, des habitudes et des schémas cognitifs;
- la redondance : il s'agit d'un recouvrement intentionnel d'informations sur les activités de l'entreprise, les responsabilités managériales et l'entreprise dans son ensemble;
- la variété requise : permet aux membres de l'organisation de composer un grand nombre de contingences.

von Krogh *et al.* trouvent le concept de gestion de la connaissance tel qu'il est entendu dans l'école technologique limité en soi. Il est important pour l'organisation de mettre en place des activateurs, ou conditions de création de connaissance :

From our perspective, managers need to support knowledge creation rather than control it [...] We call this *knowledge enabling*. (Von Krogh, Kazuo et Nonaka, 2000, p. 4)

Ainsi, les cinq conditions (p. 16) à réunir pour la création de connaissance en organisation sont :

1. instaurer une vision de la connaissance;
2. gérer les conversations;
3. mobiliser des activistes de la connaissance;
4. créer le bon contexte;
5. étendre globalement la connaissance locale.

Nonaka et Takeuchi (1997, pp. 108-113, von Krogh, Kazuo et Nonaka, 2000, pp. 82-92)) décrivent la création de connaissance comme un processus en cinq phases (voir la Figure 5) :

1. partage de connaissances tacites;
2. création de concepts;

3. justification des concepts;
4. construction d'un archétype;
5. extension de la connaissance.

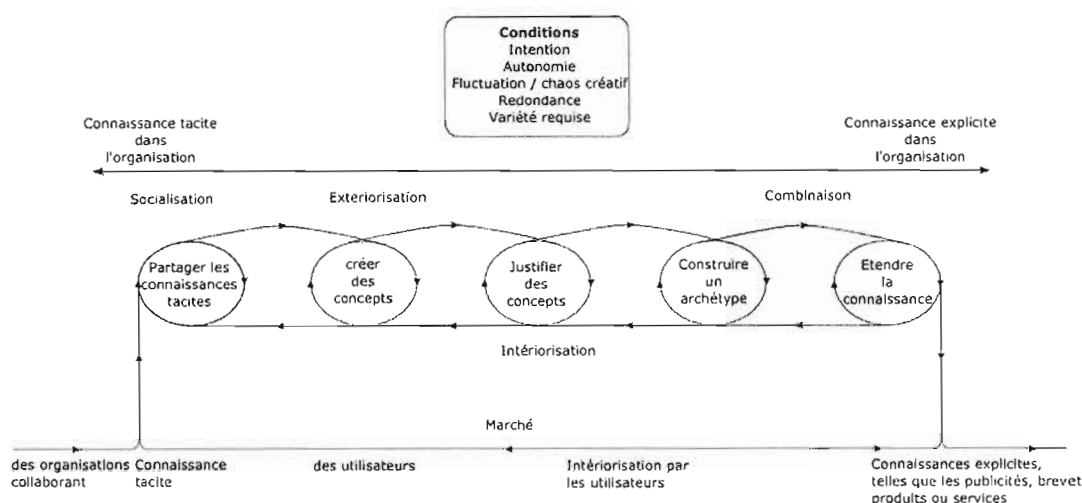


Figure 5 - Processus de création de connaissances organisationnelles (Nonaka et Takeuchi, 1997, p. 108)

1.3.4 La notion de « Ba »

Selon Nonaka et Konno (1998), la connaissance émerge dans le « Ba ». Le concept japonais de « Ba » fait référence à un espace partagé, qui peut être physique, virtuel ou mental. Le « Ba » est une plateforme d'émergence de connaissance collective. Le « Ba » est défini « *as a shared context in motion, since it is moving. Through interactions with others and the environment, both the contexts of ba and the participants grow. New knowledge is created through such changes in meanings and contexts* » (Nonaka et Toyama, 2005, p. 428). De plus, plusieurs « Ba » mis en commun forment un « Basho », où le processus de création est amplifié par rapport au « Ba ». Pour les auteurs, la connaissance sans « Ba » n'est pas de la connaissance, mais de l'information. Nonaka et Konno référencient quatre types de « Ba » :

1. l'« Originating Ba », lieu de la socialisation, permet l'échange de sentiments, d'émotions, de modèles mentaux. C'est le lieu de départ du processus de création de connaissance;
2. l'« Interacting Ba », lieu de l'extériorisation, permet la réflexion collective, afin de faire émerger la connaissance sous forme explicite;
3. le « Cyber Ba », lieu de la combinaison, est un lieu virtuel à l'abri du temps et de l'espace, il est le lieu de la raison cartésienne;
4. l'« Exercising Ba », lieu de l'intériorisation, est le lieu où la connaissance s'enracine en l'individu.

Le « Ba » donne *« l'énergie, la qualité et le lieu pour permettre à l'individu la conversion de la connaissance et le déplacement le long de la spirale de la connaissance »* (Nonaka et Toyama, 2003). L'énergie nécessaire au « Ba » pour créer de nouvelles connaissances provient des contradictions et de la pensée dialectique entre les membres.

Trois rôles doivent être joués par trois personnes différentes au sein d'un « Ba » :

- l'innovateur doit apporter une idée nouvelle;
- le coach doit apporter un point de vue extérieur, permettant d'atteindre une inter-subjectivité;
- un activiste doit voir les choses d'une manière plus large, afin d'atteindre une transsubjectivité.

1.4 La théorie de la firme créatrice de connaissance

Nonaka et Toyama (2005) tentent de répondre à la question de la théorie des firmes : pourquoi sont-elles différentes? Contrairement aux autres théories (positioning school, RBV, évolutionnistes) qui expliquent les différences comme étant le résultat de la capacité de l'entreprise à maximiser le profit, la théorie de la firme créatrice de connaissance insiste sur la capacité de la firme à tirer profit de la subjectivité des

individus, afin de façonner des visions du futur, car pour un même but, il existe différentes solutions possibles. Nonaka et Toyama en concluent la nécessité de la prise en considération de la subjectivité dans le cadre du management de l'entreprise.

De plus, l'entreprise n'est pas comme dans la théorie néoclassique une entité passive, mais au contraire, elle est dynamique, interagissant avec l'environnement et créant de la connaissance. Ainsi, il ne faut pas adopter la vision traditionnelle où la subjectivité est un bruit parasite à éliminer.

Cette croyance vient de l'épistémologie occidentale, voyant la connaissance comme une croyance vraie justifiée, laissant transparaître son objectivité. Mais pour Nonaka et Toyama, la connaissance ne peut exister sans subjectivité humaine et son contexte entourant l'humain. La Knowledge-Based Theory ne voit pas les humains comme la partie remplaçable d'une machine, mais comme des êtres différents les uns des autres.

En raison des liens entre création de connaissance et valeurs et idéaux humains, la connaissance ne peut ainsi être « capturée », comme modèle causal normatif. La théorie de la firme créatrice de connaissance ne voit pas la connaissance comme quelque chose d'absolu et d'infailible, contrairement aux visions traditionnelles, mais quelque chose de toujours incomplet, en construction.

Pour créer de nouvelles connaissances, la firme doit trouver un équilibre entre les sept valeurs suivantes (représentées sur la Figure 6) :

1. la vision, qui répond à la question « *pourquoi faisons-nous ce que nous faisons?* » (Nonaka et Toyama, 2005, p. 424), et qui contrairement à la stratégie de l'entreprise qui peut être modifiée, est une question ontologique propre à la firme, et dont la réponse n'évolue que très peu;
2. l'objectif directeur : il rend la vision concrète, en la connectant avec le processus de création de connaissance dialogues/pratiques;

3. les dialogues sont essentiels, et permettent à travers une synthèse de créer de nouvelles connaissances résolvant des contradictions. De plus, le dialogue est un outil permettant l'extériorisation et la combinaison du modèle SECI;
4. les pratiques permettent le partage de connaissances tacites entre individus (socialisation), et la conversion de connaissances explicites en connaissances tacites (intériorisation);
5. « Ba » : comme vu dans la partie 1.3.4, il s'agit du lieu de création de nouvelles connaissances. Dans la théorie de la firme créatrice de connaissances, l'entreprise peut être vue comme « *an organic configuration of multi-layered ba* » (p. 429);
6. les actifs de connaissances sont créés par le processus de création de connaissance à travers la dialectique dialogues/pratiques au sein du « Ba ». Ces actifs, contrairement aux autres types d'actifs de l'entreprise, sont spécifiques à l'entreprise, dynamiques et intangibles;
7. l'écosystème de connaissances est un « *phenomenological 'life-world'* » (p. 430), s'inscrivant dans les rapports aux sujets. Il s'agit de « Ba » multicouches à travers les frontières de l'entreprise et évoluant continuellement. En créant de nouvelles connaissances, la firme modifie perpétuellement son écosystème de connaissances.

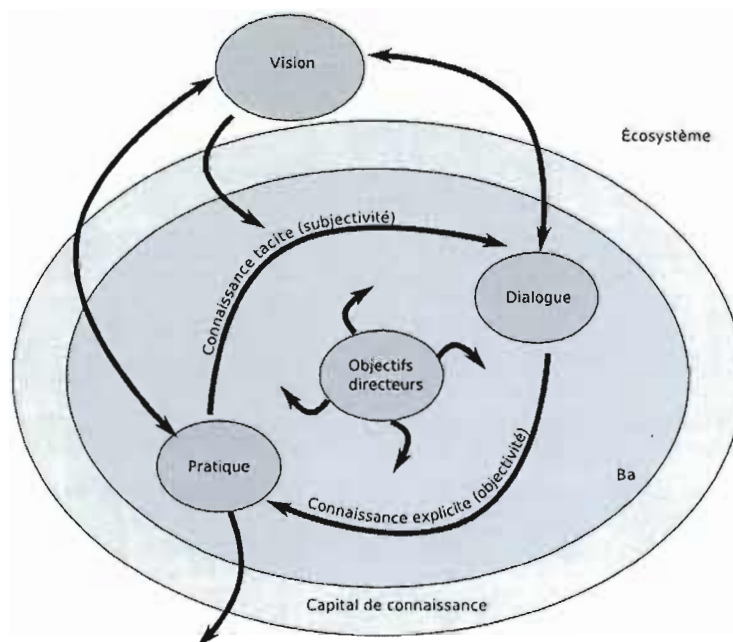


Figure 6 – Composants de la firme basée sur la connaissance (Nonaka et Toyama, 2005)

Dans la théorie de la firme créatrice de connaissances, le *leadership* consiste à être facilitateur de nouvelles créations. Ainsi, construire une vision la reprendre et la partager, construire et connecter des « Ba », les énergiser (en leur procurant des conditions comme l'autonomie, le chaos créatif, la redondance, la variété requise, ...), sont des tâches que les *leaders* doivent remplir.

1.5 Les communautés de pratique

1.5.1 Les autres formes de collectifs

Hormis les communautés de pratique (définies dans la partie 1.4.2) Prax (2000) décrit trois autres formes de collectifs dont il donne les définitions suivantes :

- la communauté d'intérêt « permet de créer et échanger des idées, opinions et croyances. Elle joue un rôle important dans la dissémination

de l'information. Ce sont par exemple des groupes de discussion, associations d'anciens élèves ... »;

- *le staff permanent « est dominé par la fonction. Il permet d'accomplir une activité permanente. C'est, par exemple, une organisation classique hiérarchique d'entreprise découpée en services, départements ou secteurs et qui va réaliser des séquences programmées en environnement stable »;*
- *le groupe projet « permet de délivrer un produit ou prestation dans un délai donné et avec des ressources affectées temporairement qui ont un rôle donné. Le flux d'informations est important. Il est un support à l'accomplissement d'une séquence de tâches pour créer collectivement de la valeur ».*

Lainesse (2003) quant à elle voit quatre types de collectifs du plus informel au plus formel: réseaux informels, communautés de pratique, communautés occupationnelles, équipe de projet ou opérationnelles.

1.5.2 La communauté de pratique : définition

Pour Forsyth (1990, p. 108), une communauté est « *un groupe d'au moins deux personnes qui s'influencent l'une l'autre à travers des interactions sociales* ».

Lave et Wenger (1991) définissent une communauté de pratique comme

a set of relations among persons, activity, and world, over time and in relation with other tangential and overlapping communities of practice. A community of practice is an intrinsic condition for the existence of knowledge, not least because it provides the interpretive support necessary for making sense of its heritage.

Lainesse (2003), définit la communauté de pratique en appuyant sur leur l'aspect communicationnel. Il s'agit pour l'auteure d'un

collectif d'individus qui interagissent et, de ce fait, tissent des liens entre eux qui leur permettent de créer et de faire circuler des connaissances relatives à un champ d'intérêt commun. Dans ce processus dynamique de mise en relation, impliquant diverses actions de communication, sont créés des 'nœuds de communication'. Ces nœuds sont le fondement même de l'engagement mutuel dans le développement d'une pratique commune. Ces nœuds de communication sont le lieu de la négociation du sens de l'action dans une entreprise conjointe. Ces nœuds sont le site d'élaboration d'une identité individuelle et collective. C'est là qu'est produit un répertoire d'objets symboliques partagés.

Plus digeste et concise, la définition que retiennent Bourhis, Dubé et Jacob d'une communauté de pratique est qu'elle est « *un groupe de personnes ayant en commun un domaine d'expertise ou une pratique professionnelle, et qui se rencontrent pour échanger, partager et apprendre les uns des autres* » (Bourhis, Dubé et Jacob, 2004).

Nous préférons dans ce mémoire la définition de Wenger et Snyder (2003) où les communautés de pratique sont définies comme « *un regroupement informel d'individus ayant en commun un domaine de spécialisation précis et une passion pour un projet collectif* ».

Wenger précise en 1998 à propos des communautés de pratique : « *it is neither a specific, narrowly defined activity or interaction nor a broadly defined aggregate that is abstractly historical and social* » (Wenger, 1998a). Pour Bourhis, Dubé et Jacob (2004), elles permettent de transcender les limites organisationnelles tels les départements, la hiérarchie, les services, ... L'une des spécificités importantes des communautés de pratique pour Wenger et Snyder (2003) est que « *ses membres s'organisent tout seuls, décident de leurs objectifs et désignent leurs responsables* ».

Il y a trois différences majeures entre communautés de pratique et « Ba » (Nonaka et Toyama, 2003, p. 7) :

1. la communauté de pratique est le lieu où les membres apprennent la connaissance qui est dans la communauté, alors que le « Ba » est un lieu de création nouvelle de connaissance;
2. la communauté de pratique a une identité, une frontière, une culture, une histoire alors que le « Ba » a des frontières changeantes et fluides, et peut changer rapidement, ainsi que ses participants;
3. les membres d'une communauté de pratique sont assez stables, cela prend un certain temps aux nouveaux participants pour être pleinement intégrés, alors que les membres d'un « Ba » ne sont pas fixés, ils vont et viennent.

Toutefois, « Ba » et communauté de pratique ne sont pas incompatibles, et un « Ba » peut très bien se former au sein d'une telle communauté.

1.5.3 Les niveaux de participation dans une communauté de pratique

Dans une communauté de pratique, plusieurs niveaux de participation coexistent. Lave et Wenger (1991) ont introduit le concept de « *legitimate peripheral participation* » afin de comprendre le processus de socialisation et d'apprentissage dans une communauté de pratique. Il s'agit de l'implication progressive des nouveaux venus dans la communauté existante au fur et à mesure qu'ils acquièrent les compétences nécessaires à la pratique. Les nouveaux membres passent alors d'une participation périphérique à une pleine participation dans la communauté.

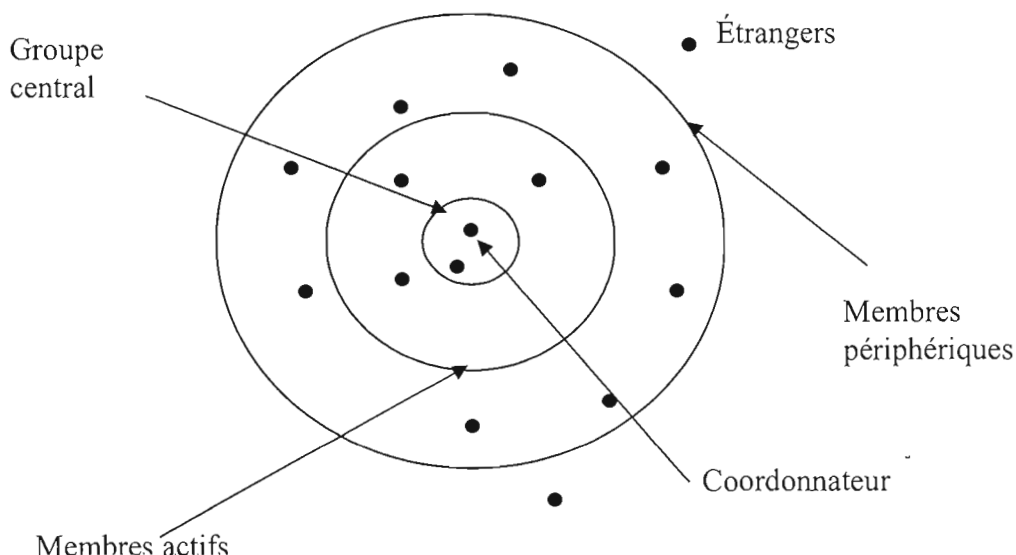


Figure 7 - Niveaux de participation dans une communauté de pratique (d'après Wenger, McDermott et Snyder (2002))

Wenger, McDermott et Snyder (2002) décrivent aussi un cœur, des membres actifs et d'autres périphériques. Ils considèrent normal le fait de passer d'un degré à un autre de participation. McDermott et Jackson (2002) décrivent les trois niveaux de participation :

A small core group of people actively participates in discussions, even debates, in public community forum. They take on community projects, identify topics for the community to address, and move the community along its learning agenda. [...] Active members make up the second level of participation. These members attend meetings regularly and contribute occasionally to the community forums. [...] Most community members rarely participate. Instead they lurk at the periphery of the community, watching the interactions of the core and active members. [...] Community members change levels of participation [...].

Wenger et al. (2002) ajoutent qu'à l'extérieur de ces principaux niveaux de participation, on retrouve des gens – nommés étrangers – qui ne sont pas membres de la communauté, mais qui lui portent un intérêt.

La participation permet le partage d'expérience et de connaissances. Ce partage est au cœur du concept de communauté de pratique : « *les gens impliqués dans ces groupes*

partagent expérience et connaissance, librement et avec une créativité qui favorise le développement de nouvelles approches des problèmes ». (Wenger et Snyder, 2003). Ceci découle de la vision de l'apprentissage en tant que phénomène social. Les membres potentiels ou effectifs d'une communauté de pratique sont ainsi guidés par le réel désir et besoin de partager et de connaître.

Selon Wenger (1998a), les communautés de pratique peuvent être comprises comme étant des histoires partagées d'apprentissage. Ce processus dynamique de partage et d'apprentissage de nouveaux savoirs est l'endroit où est construite la signification des choses, où s'effectue la négociation du sens.

1.5.4 Le rôle des communautés de pratique

Pour développer le rôle de la communauté de pratique, Ballay (2002, p. 164) décrit la compréhension commune de l'entreprise actuelle comme étant une matrice à deux dimensions : une logique hiérarchique et une logique client fonctionnant plus par projet (voir Figure 8).

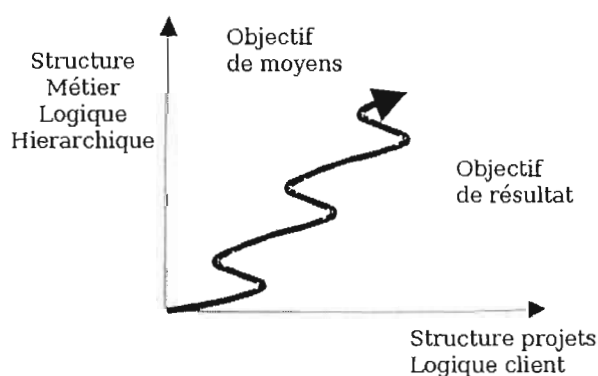


Figure 8 - L'organisation matricielle et ses logiques d'évaluation, d'après Ballay (2002, p. 166)

Pour Ballay, la communauté de pratique à travers la logique du savoir est la dimension manquante à la structure matricielle (logique hiérarchique verticale / logique client horizontale) pour que celle-ci fonctionne effectivement (voir Figure 9).

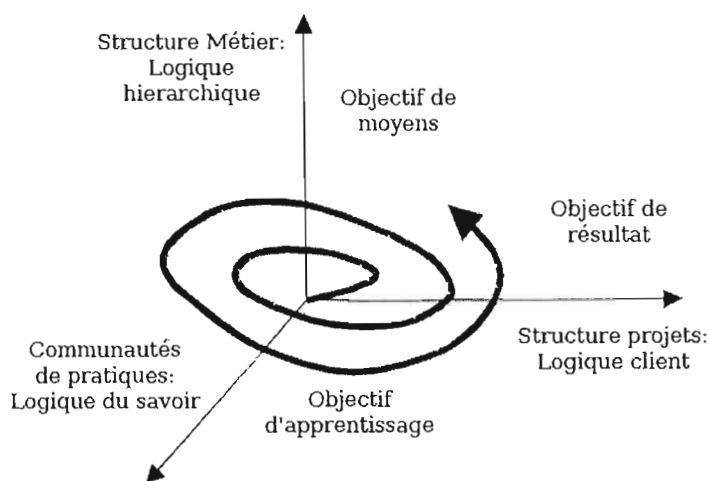


Figure 9 - Les communautés de pratique – la dimension du savoir, d'après Ballay (2002, p. 167)

1.5.5 Les effets explicites des communautés de pratique

Pour Wenger et Snyder (2003), les communautés de pratique ont pour effets directs de :

- donner un coup de fouet aux efforts stratégiques;
- ouvrir de nouveaux marchés;
- résoudre des problèmes rapidement;
- accélérer la diffusion des meilleures pratiques;
- contribuer à l'acquisition de compétences professionnelles;
- aider l'entreprise à recruter et fidéliser des talents.

1.5.6 Les communautés de pratique virtuelles

Selon Dubé et *al.*, une communauté de pratique est dite virtuelle quand le moyen principal d'interaction entre ses membres est du domaine des TIC (Dubé, Bourhis et Jacob, 2003). Pour Wenger et *al.*, lorsque les communautés ne peuvent compter sur l'interaction physique de ses membres, il s'agit de communautés dites *distributed* (Wenger, McDermott et Snyder, 2002). Les interactions à distance à travers les TIC

ne pouvant se substituer aux contacts réels, une moindre présence des membres est généralement observée. Cela nécessite un effort plus important quant au maintien des liens. Toutefois, les communautés sont généralement plus grandes et culturellement plus variées, puisqu'elles sont distribuées sur un plus vaste territoire, ce qui n'est pas sans créer de nouveaux obstacles. Toutefois, selon Bourhis et *al.*, la satisfaction des membres de ces communautés est très importante (Bourhis et Tremblay, 2004, pp. 66-67).

Dans ce chapitre, nous avons traité du passage de l'économie traditionnel à une économie où la connaissance est de plus en plus importante, avec un rôle primordial donné aux TIC. Nous nous sommes intéressés à ce qu'était la connaissance, et découlant de la conception que l'on s'en fait, des différentes écoles de la gestion des connaissances. Nous avons présenté le « Ba », la théorie de la firme créatrice de connaissance, ainsi que les communautés de pratique. Dans le chapitre suivant, nous nous intéresserons à l'univers du logiciel.

CHAPITRE II

REVUE DE LITTÉRATURE - LOGICIELS ET LOGICIELS LIBRES

Dans ce chapitre, nous traiterons de l'univers du logiciel. Dans un premier temps, nous donnerons quelques définitions pour comprendre le milieu informatique, et à travers l'économie du logiciel, nous traiterons de ses propriétés, ses modes de production, ... Puis, nous traiterons de la propriété intellectuelle qui possède quelques spécificités dans le logiciel, à la suite de quoi nous aurons les outils conceptuels afin de nous permettre d'aborder les logiciels libres, ainsi que leur modèle de production qui n'est pas sans rappeler le modèle décrit par Mauss (1923-1924) d'économie du don. Nous verrons toutefois que l'univers du libre est en mutation et que la notion de libre s'étant à d'autres domaines que le logiciel. Nous examinerons ensuite les logiciels libres, et traiterons de l'innovation et de la gestion des connaissances dans cet univers.

2.1 Le logiciel : définition

Pour Caillaud, un programme est « *une suite d'instructions, écrites dans un langage bien défini, qui codent des opérations, des procédés ou des algorithmes s'appliquant à des données au sens large, à de l'information* » (Caillaud, 2003, p. 115). Quant au logiciel, l'auteur le définit comme « *un ensemble de programmes (ou sous-programmes) qui interagissent* ». Horn précise que le logiciel appartient à un système informatique, et que le système informatique a pour fonction de traiter de l'information. D'ailleurs, le mot « informatique » vient de la compression des mots « information » et « électronique » (Horn, 2000, p. 15). Le système informatique en soi traite non pas de l'information, car cette dernière nécessite d'être interprétée a

priori, mais des données, qui ne deviendront de l'information que pour un acteur humain (p. 36). Ainsi, flux de données, d'informations et de connaissances se répartissent tels que présentés sur la Figure 10.

Caillaud voit trois types différents de logiciels :

- les logiciels embarqués, directement liés à des systèmes physiques (ex. : contrôleur de machine, d'avion, ...);
- les logiciels d'infrastructures, utilisés non pas directement par l'utilisateur, mais par d'autres programmes (ex. : systèmes d'exploitation, logiciels de compression de données, de transmission TCP/IP, ...);
- les programmes d'application, utilisés directement par l'utilisateur (ex. : traitement de texte, retouche d'images, calcul numérique, ...).

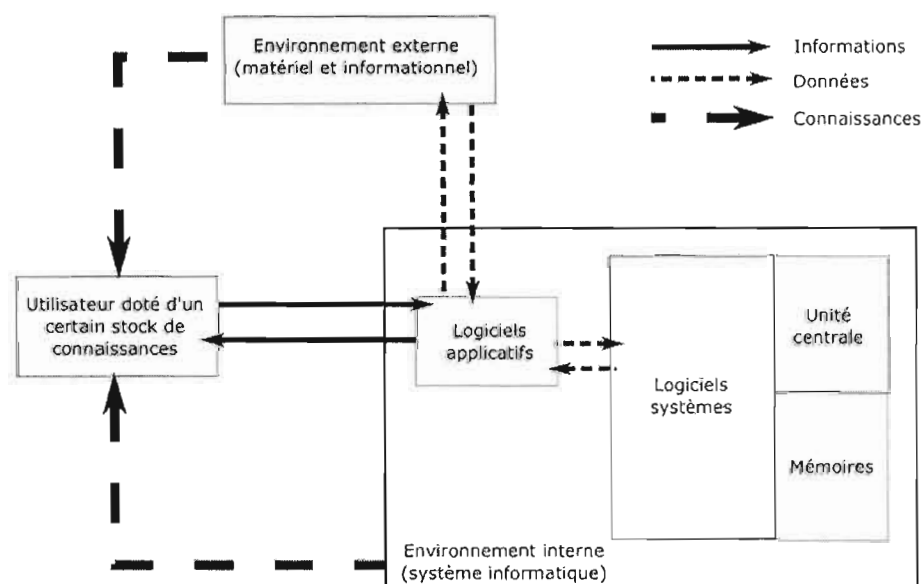


Figure 10 - Flux de données, d'information et de connaissance autour des systèmes informatiques (d'après Horn, p.38)

Horn (p. 40) décrit trois dimensions du logiciel :

- le logiciel comme texte numérique actif;
- le logiciel comme création humaine qui devient un « acteur non humain »;

- le logiciel comme « objet frontière ».

2.2 L'économie du logiciel

L'économie du logiciel est confrontée à un environnement bien particulier, dans un environnement où la puissance matérielle informatique double tous les 18 mois, d'après la « loi de Moore » (Chollet, 2004, p. 8, Horn, 2000, p. 112). D'un point de vue qualitatif, le type même de matériel change lui aussi, les logiciels devant en tenir compte. L'innovation en termes de logiciels concerne l'apparition de nouveaux produits, de nouvelles méthodes de production, l'utilisation de nouveaux outils. Les changements sont extrêmement rapides, ne laissant pas le temps à l'utilisateur de les assimiler (Horn, 2000, p. 106). L'instabilité technologique est la norme. De plus, la distinction bien – service s'applique difficilement (p. 132). Les lois sur la propriété intellectuelle sont sujettes à de multiples changements, et sont instables dans le temps et dans l'espace (voir la section suivante). Les procédés classiques pour déterminer la valeur – c'est-à-dire suivant la valeur d'échange, donc sur le marché – ne sont pas applicables, de par la nature même du logiciel. La valeur d'usage est quant à elle difficile à déterminer. Enfin, notons que pour Pesqueux, « *l'informatique et plus généralement les TIC renvoient bien à un idéal organisationnel véhiculé par le thème du réseau* » (2002, p. 205).

Caillaud décrit le mode de production du programme comme étant d'abord issue d'un langage de programmation, langage compréhensible pour l'homme, sous la forme d'un code source. Puis ce code source est transformé en une suite binaire de 0 et de 1, compréhensible pour la machine – c'est le code binaire, ou code objet. L'utilisation du programme binaire ne nécessite pas la transmission du code source à partir duquel celui-ci a été construit. Horn (p. 45) décrit le logiciel comme la résultante des connaissances des producteurs, des connaissances des utilisateurs, des outils et logiciels (autres) et matériels.

Horn (ch. VII) propose une typologie des mondes de productions en fonction des types de produits et de la présence ou non de standardisation (voir Tableau 1). Le monde interpersonnel est celui de la production sur-mesure et du service personnalisé. Le monde fordiste est celui de la production de masse. Le monde de la création est celui de la recherche, des scientifiques, propre au développement de nouvelles connaissances. L'utilisateur y a peu d'influence sur la production. Enfin, le monde de la production flexible prend la forme de produits individualisés fabriqués à partir de composants et/ou méthodes standardisées, et de composés dont certaines parties sont normalisées.

Tableau 1 - Typologie des mondes de productions

	Absence de standardisation	Standardisation du produit et/ou des composants
Produits dédiés	Monde interpersonnel	Monde de la production flexible
Produits génériques	Monde de la création	Monde fordiste

Traditionnellement, les logiciels sont écrits par des *hackers*. De nombreux auteurs (Raymond, Torvalds, Himanen, ...) distinguent le *hacker* du *cracker*. Les *crackers* sont des gens « *qui prennent leur pied en s'introduisant dans les ordinateurs et en piratant le réseau téléphonique* » (Raymond, 2000). Pour Raymond, un *hacker* est une personne appartenant à « *une communauté - une culture partagée - de programmeurs chevronnés et de sorciers des réseaux* ». Un *hacker* appartient à cette culture et peut se considérer *hacker* si d'autres personnes ayant contribué à cette culture le reconnaissent comme tel. Pour Himanen, être *hacker* relève d'un état d'esprit, une liberté de pensée, et un plaisir. Il cite Burrell Smith¹¹ : « *les hackers peuvent faire n'importe quoi et être hackers. Vous pouvez être un charpentier hacker. Il n'est pas indispensable d'être à la pointe des technologies. Je crois que cela a à*

¹¹ À l'origine du Macintosh d'Apple.

voir avec l'art et le soin qu'on y apporte »¹² (Himanen, 2001, p. 26). Pour Linus Torvalds¹³, la source de motivation des *hackers* est le plaisir. « *l'ordinateur lui-même est source de plaisir* » (Torvalds, 2001, p. 18). Pour Himanen, la conception protestante du travail voit le travail comme une fin en soi, et justifie le capitalisme et l'accumulation de profit. Mais, le *hacker* possède une éthique différente, centrée sur trois pôles :

- le travail est vu comme un jeu, une passion;
- l'argent est moins important que le partage;
- le réseau rend la structure plate (Himanen, 2001).

2.3 La propriété intellectuelle dans le logiciel

Chabchoub décrit le système de propriété intellectuel dans le logiciel comme relevant à la fois du droit d'auteur, du secret commercial, et des brevets. Au Canada, le droit d'auteur est basé sur trois principes : la structure, la séquence et l'organisation. La structure représente la définition de la relation entre différents modules et sous-programmes, la séquence correspond à l'ordre de déroulement des différentes parties, et l'organisation est la définition des fonctions de chaque module ou sous-programme (Chabchoub, 2005, p. 78). Les brevets peuvent protéger le logiciel s'ils sont formulés en termes de problème technique portant sur l'invention technique d'un procédé. Il s'en suit que « *l'algorithme d'un logiciel peut être considéré comme le procédé qui décrit le logiciel* » (p. 79), d'où la brevetabilité de l'algorithme. Notons toutefois que le brevet sur les logiciels est fortement contesté, car il cherche à rétablir artificiellement la rareté dans un milieu d'abondance, sans privilégier la véritable rareté : la créativité du programmeur (Lang, 2000). Pour Lang, l'abus de propriété intellectuelle visant la recherche de monopole forme « *des barrières bien plus novices*

¹² Levy, Hackers, p.434.

¹³ Programmeur à l'origine du noyau Linux, et encore actuel architecte principal du noyau,

au progrès économique et technique et à la création d'emplois utiles ». C'est la raison pour laquelle Lang qualifie sur cette question le libéralisme économique d'hémiplégique.

Raymond pose la pertinente question du sens de la propriété :

Que signifie le terme « propriété » lorsque ce qui est possédé est duplicable à l'infini, hautement malléable, et que la culture environnante n'est ni capable de faire appliquer des lois, ni dans une situation économique où les ressources sont limitées ? (Raymond, 1999)

Pour Raymond, les propriétaires, dans le cas des logiciels open source, sont ceux qui ont le droit exclusif d'en redistribuer des versions modifiées.

Pour Stallman, le logiciel ne devrait pas avoir de propriétaire. Car la possibilité de copie apporte de multiples avantages (Stallman, 2000). Pour Quéau :

Il faut revenir aux intuitions premières. Il est plus avantageux pour l'humanité de faire circuler librement les idées et les connaissances que de limiter cette circulation. Aristote affirme dans la *Poétique* que l'homme est l'animal mimétique par excellence. Pour lui la *mimesis* est créatrice de modèles. Les Lumières reprirent le concept d'imitation. Pour Condillac, « les hommes ne finissent par être si différents, que parce qu'ils ont commencé par être copistes et qu'ils continuent de l'être ». La copie ajoute une aura à la chose copiée. Pour le philosophe Alain, « copier est une action qui fait penser » (Quéau, 2000).

Toutefois, on assiste à de fortes modifications dans le monde de la propriété intellectuelle du logiciel, en allant vers de plus en plus de brevets (Lang, 2000). Pour Blondeau

Ce brouillage du rapport capital/travail sur ces deux aspects de la question de la propriété incite le capital à opérer un rééquilibrage qui s'inscrit tout entier dans l'expression juridique des rapports de propriété. Avec la déstabilisation de la condition salariale comme projet de société et l'exode d'une partie du capital vers la spéculation financière¹⁴, le renforcement, sinon le verrouillage, de la

¹⁴ André Gorz, *Misère du présent, richesse du possible*, Galilée, Paris, 1997

propriété intellectuelle sur la marchandise est en effet une des principales composantes de la stratégie du capital (Blondeau, 2000a).

2.4 Le logiciel libre : théories et valeurs

2.4.1 Logiciel libre et logiciel open source : définition

Le logiciel libre est défini par la FSF¹⁵ comme devant respecter les quatre libertés suivantes :

0. la liberté d'exécuter le programme, pour tous les usages.
1. la liberté d'étudier le fonctionnement du programme, et de l'adapter à vos besoins. Pour ceci l'accès au code source est une condition requise.
2. la liberté de redistribuer des copies, donc d'aider votre voisin.
3. la liberté d'améliorer le programme et de publier vos améliorations, pour en faire profiter toute la communauté. Pour ceci l'accès au code source est une condition requise (FSF, 2004).

Pour Stallman,

L'idée que le système social du logiciel propriétaire — le système qui vous interdit de partager ou d'échanger le logiciel — est antisocial, immoral, et qu'il est tout bonnement incorrect, surprendra peut-être certains lecteurs. Mais comment qualifier autrement un système fondé sur la division et l'isolement des utilisateurs ? Les lecteurs surpris par cette idée ont probablement pris le système social du logiciel propriétaire pour argent comptant, ou l'ont jugé en employant les termes suggérés par les entreprises vivant de logiciels propriétaires (Stallman, 1999).

La différence de liberté entre utilisateur et distributeur est exprimée par Stallman ainsi :

‘Ce programme est-il libre ?’ Qui fondait son jugement sur la liberté fournie par les conditions de distribution de la distribution du *MIT*, concluait que *X* était un

¹⁵ La FSF est la Free Software Foundation, créée par Richard Stallman

logiciel libre. Mais qui mesurait la liberté de l'utilisateur type de *X*, devait conclure que *X* était un logiciel propriétaire. La plupart des utilisateurs de *X* exécutaient des versions propriétaires fournies avec des systèmes *Unix*, et non pas la version libre (Stallman, 1999).

Pour Raymond, le logiciel open source est un logiciel dont le code source est disponible. Ce type de logiciel permet le mode de développement en « bazar » (distribution rapide et fréquente, délégation importante, utilisateurs considérés comme codéveloppeurs), par opposition au mode de développement en « cathédrale », où seuls peu d'individus disposent du code source entre deux diffusions (Raymond, 1998). Pour Samudrala, *« la technologie des logiciels librement redistribuables n'est pas un nouveau concept [...] en revanche, le fait que ce type de logiciels fasse naître un secteur commercial est tout à fait nouveau »* (Samudrala, 2000). Ainsi,

En libérant la puissance créatrice des milliers de programmeurs qui opèrent sur Internet et en intégrant leurs meilleurs enrichissements aux futures versions du Communicator [de Netscape], nous pensons pouvoir amener les innovations technologiques sur le marché des navigateurs à un niveau d'excellence jamais atteint¹⁶.

Le logiciel à code source ouvert doit – pour l'OSI (Open Source Initiative)¹⁷ – être librement redistributable, inclure le code source, autoriser les travaux dérivés, respecter l'intégrité de l'auteur du code source, ne pas discriminer des personnes ou des groupes, ne pas discriminer des champs de préoccupation, attacher la licence au logiciel, ne pas être spécifique à une partie du logiciel, ne pas restreindre d'autres logiciels, être technologiquement neutre (OSI, 1997-2006).

Pour Stallman, la différence entre logiciel libre et logiciel open source est que

la rhétorique de l'« Open Source » met l'accent sur le potentiel pour faire du logiciel puissant et de grande qualité, mais fait passer au second plan les idées de liberté, de communauté, et de principes[...] Les termes « Free Software » et

¹⁶ J. Barksdale, «Software development for the greater good» <http://home.netscape.com/comprod/columns/mainthing/source.html?cp=nws04flh4>.

¹⁷ L'OSI est l'organisme promouvant le logiciel open source, organisme créé par Eric S. Raymond

« Open Source » décrivent tous deux plus ou moins la même catégorie de logiciels, mais correspondent à des conceptions différentes du logiciel et des valeurs qui lui sont associées (Stallman, 1999).

Stallman produit un contre-argumentaire aux cinq arguments qu'il voit selon lesquels les logiciels devraient avoir un propriétaire :

- les insultes, qui conduisent à faire des analogies simplistes pour décrédibiliser les opposants à la propriété sur les logiciels;
- l'exagération, où les coûts de la contrefaçon informatique sont calculés en fonction du prix « non gagné » de chaque logiciel contrefait, alors que seule une minorité aurait vraiment été achetée;
- le droit : il est possible selon la loi d'avoir des droits de propriété intellectuelle sur un logiciel, mais le droit est le reflet d'une construction et d'une volonté humaine;
- les droits naturels : le thème des droits naturels a déjà été abordé à la section 1.1.4;
- l'économie : la vente du logiciel serait plus rentable globalement pour la société à travers un plus grand nombre d'innovation. Mais, comme nous le verrons à la section suivante sur l'économie du don, cette dernière est plus efficace que l'économie traditionnelle dans le domaine de l'intangible. (Stallman, 2000)

Pour lever l'ambiguïté entre logiciel libre et logiciel à code source ouvert, on trouve aussi dans la littérature les termes F/OSS pour Free / Open Source Software et FLOSS pour Free/Libre Open Source Software (Wheeler, 2005). Couture voit dans l'opposition « Logiciel Libre » / « Logiciel à code source ouvert » une controverse d'ordre sociotechnique au sens de Latour, et c'est par l'étude de ces controverses que se fait la construction des différents modèles du libre, car pour l'auteur le libre n'est pas encore stabilisé (Couture, 2006).

2.4.2 Le logiciel libre dans l'économie

Dans une économie du logiciel libre, les fournisseurs de logiciels peuvent se transformer en fournisseurs de services plutôt que de produits (Stallman, 1999) : le logiciel est disponible, mais si des entreprises ou individus veulent un support spécifique répondant à leur besoin, l'entreprise travaillant dans le logiciel libre peut lui fournir.

Pour Weber, les revenus des logiciels libres peuvent provenir de la marque associée à ces logiciels, qui tirent ainsi profit de l'effort effectué en développement pour leur renommée et leur savoir-faire. Les revenus peuvent aussi être faits sur les connaissances tacites dans le fonctionnement des logiciels et du monde du libre des individus de l'entreprise, pour transformer le code source disponible en solutions techniques pour des clients (Weber, 2004).

2.4.3 Les adversaires du libre

De par ses modes de développements et sa remise en cause des modèles traditionnels, tous ne voient pas favorablement le libre. Ainsi, le *think tank* Alexis de Tocqueville Institution critique régulièrement le logiciel libre pour des raisons de sécurité¹⁸, de révisionnisme historique¹⁹ (qualifiant le logiciel open source de route vers la « *raymondization* »), de menteurs²⁰, de communistes²¹, de voleurs de code²², etc.²³.

¹⁸ Source : <http://adti.net/defense/security.update.html>

¹⁹ Source : <http://adti.net/samizdat/opensores.html>

²⁰ Source : http://news.zdnet.com/2100-3513_22-5216651.html

²¹ Source : <http://blogs.zdnet.com/open-source/index.php?p=210>

²² Source : « Intellectual property — left? », <http://www.adti.net/ip/laches.050405.pdf>

²³ Les articles et rapports de l'institution sont disponibles sur la page <http://www.adti.net/gw-intellprop.html>

2.4.4 Le logiciel libre : une économie du don ?

Dans son essai sur le don, Marcel Mauss met en lumière une forme de fonctionnement de la société différent des systèmes capitalistes et communistes, qu'il qualifie de phénomènes sociaux « totaux ». Pour Mauss, cette forme de fonctionnement est un phénomène proprement humain, se cachant derrière le fonctionnement traditionnel de nos économies modernes (Mauss, 1923-1924).

Selon Raymond, l'univers informatique est un univers où il y a abondance (espace disque, réseau, bande passante, puissance de calcul), propre à une culture du don, c'est-à-dire un espace où le statut social n'est pas déterminé par ce qu'on contrôle, mais par ce qu'on donne, contrairement aux deux autres modèles : (1) le pouvoir centralisé où le statut social est déterminé par l'accès à un pouvoir répressif, et (2) l'économie d'échange où le statut social est déterminé par le contrôle que l'on a sur les marchandises (Raymond, 1999).

De même, pour Barbrook, l'économie du don, soutenue par l'idéologie anarcho-communiste qui, contrairement à l'aliénation de la société bourgeoise, permet le contact intime et l'authenticité affective, et donc vue comme l'antithèse de l'économie de marché s'est insinuée au cœur même de cette économie, d'abord dans le microcosme universitaire, puis à travers ce canal dans le fonctionnement d'Internet. Ainsi, « *la structure du Net rend donc la propriété intellectuelle techniquement et socialement obsolète* ». Ainsi, « *dans l'économie du don high-tech, les gens travaillent ensemble avec succès grâce à "un processus social ouvert incluant évaluation, comparaison et collaboration"*²⁴ ». Pour l'auteur, ce n'est pas la fin de l'histoire²⁵, mais au contraire l'économie du don – née au cœur de la plus puissante économie capitaliste – est devenue « *la seule alternative à la domination du*

²⁴ Bernard Land, « *Des logiciels libres pour tous* », le monde diplomatique, janvier 1998, p.26

²⁵ Référence à Francis Fukuyama, « *La fin de l'histoire et le dernier homme* »

capitalisme monopolistique ». Toutefois, loin de s'affronter, ces modèles différents se complètent, et coexistent en symbiose.

L'économie du don et le secteur commercial ne peuvent se développer qu'en s'associant au sein du cyberspace. Le libre échange de l'information entre les utilisateurs s'appuie sur la production capitaliste d'ordinateurs, de logiciels et de télécommunications. Les bénéfices des firmes commerciales sur le Net dépendent de l'augmentation du nombre de gens qui participent à l'économie du don *high-tech* (Barbrook, 2000).

Ainsi, selon Blondeau, « *la Noosphère est en passe de concurrencer le marché sur son propre terrain : celui de la rentabilité et de l'efficacité* » (Blondeau, 2000b).

Pour Stallman, l'idée selon laquelle les sociétés éditrices de logiciel disposeraient d'un droit naturel, inaliénable, à posséder le logiciel est fausse, car « *la constitution et la tradition juridique des États-Unis d'Amérique rejettent toutes deux cette idée; le copyright n'est pas un droit naturel, mais un monopole artificiel, imposé par l'État, qui restreint le droit naturel qu'ont les utilisateurs de copier le logiciel* » (Stallman, 1999).

2.4.5 Les structures du libre

Bien que Raymond décrive un modèle de logiciel *open source* largement repris (Raymond, 1998), Crowston et Howison à travers une étude sur 120 projets hébergés par SourceForge montrent qu'il n'y a pas un type de structure, mais une multitude de structures possibles dans le libre (Crowston et Howison, 2005).

Dans leur étude sur le projet SPIP, Demazière *et al.* font ressortir une structure en trois cercles polycentriques : un cercle stratégique restreint, un deuxième cercle de contributeurs actifs et un troisième cercle périphérique. Le cercle de contributeurs « *faisant office d'intermédiaires entre le cercle périphérique et stratégique* » (Demazière, Horn et Zune, 2006, p. 5).

2.4.6 Les incitations à la contribution dans le logiciel libre

Dans le projet SPIP, Demazière *et al.* déterminent trois types de motivation pour le cercle intermédiaire : le militantisme politique, l'attrait technologique et les professionnels tentant de développer un modèle économique sur le logiciel (Demazière, Horn et Zune, 2006).

Auray et Vicente voient le logiciel libre comme une réaction au moment intégré, se positionnant « *dans l'action et dans une quête existentielle qui va au-delà du travail* » (Auray et Vicente, 2006 4). Pour les individus contribuant aux projets étudiés (Debian et OpenBSD), l'engagement dans le logiciel libre correspond

à un désintéressement ou une déception vis-à-vis de l'activité professionnelle actuelle, sur laquelle les participants n'hésitent pas à émettre des critiques virulentes [...] majoritairement sur quatre points : la temporalité, la proximité, la qualité et le rapport à l'objet produit (Auray et Vicente, 2006 5).

Pour Foray et Zimmermann, les mécanismes du logiciel libre engendrent des incitations de deux ordres :

- une incitation collective, l'effort collectif d'amélioration entraîne pour chacun une externalité d'usage;
- des incitations individuelles : les contributeurs ont des effets d'apprentissage (par la pratique du codage, et par la lecture du code d'un autre), et de réputation, entraînant une forme de reconnaissance par les pairs (Foray et Zimmermann, 2001, pp. 12-13).

2.4.7 Une mutation de l'univers du libre

Foray et Zimmermann produisent un modèle où l'apprentissage et les gains de réputation baissent avec la forte augmentation du nombre d'utilisateurs de logiciel libre, éliminant ainsi les incitatifs individuels présentés. Toutefois, ils tempèrent cette

analyse par le niveau de l'engagement militant des individus (Foray et Zimmermann, 2001, p. 17).

Ainsi, le logiciel libre subirait aujourd'hui une mutation de son écosystème, due à agrandissement du nombre d'utilisateurs non-développeurs, et donc une réduction relative d'utilisateurs-développeurs (Jullien et Zimmermann, 2005). L'enjeu pour les auteurs est alors de savoir capter l'énergie des nouveaux utilisateurs, pour la transformer positivement pour le projet et ne pas tuer les communautés par non-respect de la culture *hacker* y existant.

2.4.8 Une extension de la notion de libre

Il est intéressant de noter que le libre se diffuse à d'autres domaines sous différentes licences comme GFDL ou Creative Commons (Goldenberg, 2006), où encore l'encyclopédie en ligne Wikipedia et les autres sites s'y rapportant (Molinié, 2006). A noter aussi, des projets de livre d'éducation libre²⁶, de création de matériel informatique libre par de grandes entreprises telles Sun Microsystems²⁷, ou encore le fait de parler de génomique open source pour des biotechnologies SAM (Sélection Assistée par Marqueurs) se voulant être une technique plus efficace et plus sûre (pour l'environnement et l'humain) que les OGM (Organismes Génétiquement Modifiés)²⁸.

²⁶Source : « Sortie du premier manuel scolaire libre - Interview Sésamath », <http://www.framasoft.net/article339.html>

²⁷ Source : « SUN libère ses processeurs SPARC », <http://linuxfr.org/2005/12/13/20055.html>

²⁸ Source : « Au-delà des OGM », Jeremy Rifkin, Libération du mardi 18 juillet 2006.

2.5 Les logiciels libres

Il existe 124 980 projets libres hébergés par SourceForge²⁹. Freshmeat.net recense quelque 74 231 projets libres³⁰, répartis tel que présenté sur la Figure 11.

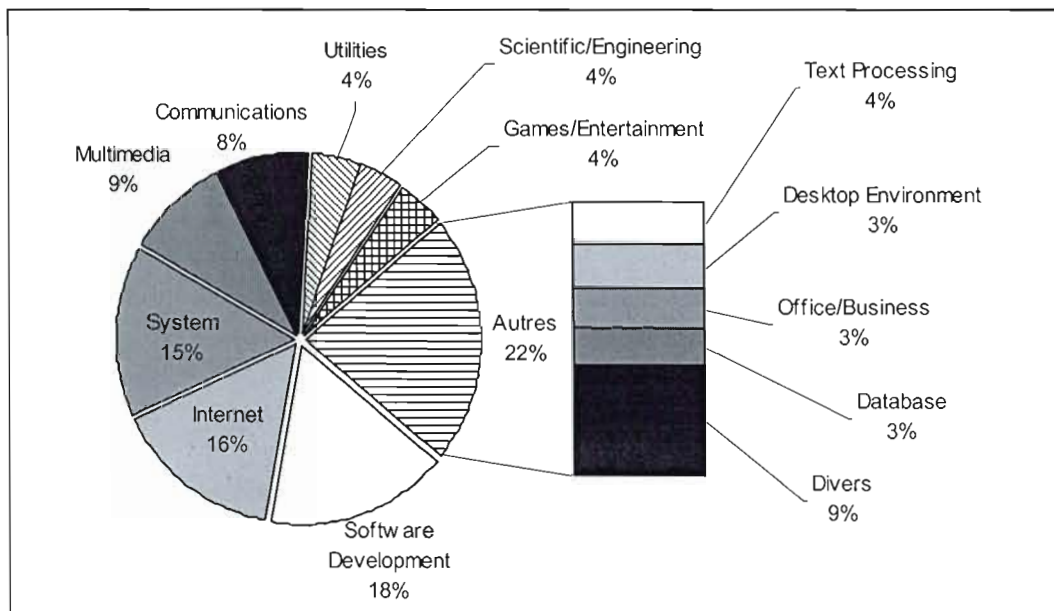


Figure 11 – Répartition des différentes catégories de projets libres, d'après freshmeat.net

Sur un total de 98502 projets recensés entre 2001 et 2005 sur SourceForge, à peine 2% ont eu une équipe avec un maximum de dix personnes ou plus (voir Figure 12) (Crowston, Howison et Annabi, 2006, p. 137).

Dans cette multitude de logiciels libres, quelques grands projets sont plus connus et populaires que les autres. Citons, à titre non exhaustif : Firefox (navigateur web), Thunderbird (lecteur de courriel), Linux (noyau de système d'exploitation),

²⁹ Source : <http://sourceforge.net/>, le 20 juillet 2006. SourceForge est le plus gros hébergeur de projet libre sur internet.

³⁰ Source : <http://freshmeat.net/browse/18/>, le 20 juillet 2006. Freshmeat.net est un index de programmes, en très grande majorité libres.

Openoffice (suite bureautique), apache (serveur web), Mysql (système de gestion de base de données), PHP (langage de programmation).

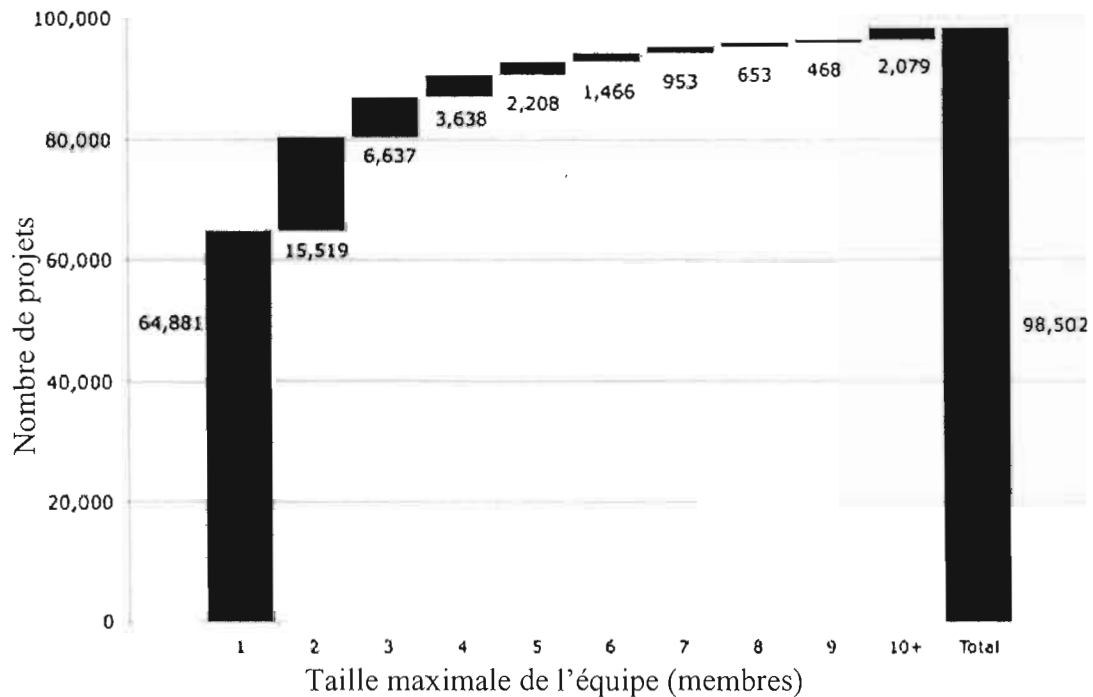


Figure 12 - Nombre maximum de développeurs listés par projet sur SourceForge sur une période de cinq ans depuis 2001. (Source : Crowston, Howison et Annabi)

Il existe un grand nombre de licences libres, mais les quatre principales sont :

- les licences BSD & MIT : il est possible d'incorporer du code propriétaire et redistribuer le projet sans distribuer les nouvelles modifications;
- les licences General Public License (GPL), possédant une gauche d'auteur³¹ "*Copyleft*" : en distribuant le logiciel, il est nécessaire de distribuer le code source avec. Il n'est pas possible d'embarquer dedans du code propriétaire;

³¹ Le terme « gauche d'auteur » est la traduction du *copyleft* qui s'oppose au *copyright*, ou « droit d'auteur »

- les licences Lesser/Library GPL (LGPL) : un compromis entre les deux précédentes licences : le code source doit être distribué, mais il est possible d'embarquer du code propriétaire;
- les logiciels dans le domaine public sont aussi des logiciels libres (Wheeler, 2005).

En plus de la liberté et de l'efficacité des logiciels libres, Wheeler dénombre un ensemble d'arguments à l'utilisation des logiciels libres : de meilleures performances, une plus grande indépendance matérielle, une meilleure sécurité, un coût inférieur, un suivi des données sur le long terme (le format dans lequel les données sont enregistrées est documenté), etc. (Wheeler, 2005).

2.6 L'innovation et la gestion des connaissances dans les logiciels libres

En analysant l'économie de l'industrie du logiciel et de l'informatique, Bessen et Maskin, y caractérise l'innovation comme séquentielle (une nouvelle avancée s'appuie sur les précédentes) et complémentaire (plusieurs recherches dans un même but, mais avec différentes approches augmentent la probabilité d'aboutissement). Ils en concluent qu'un système de protection trop fort frêne l'innovation (Bessen et Maskin, 1999).

Pour von Hippel le modèle du libre correspond au modèle d'innovation ascendante ou innovation par l'usage, où la catégorie d'utilisateurs-innovateurs, en usant de sa créativité, innove à son échelle en redisant les choses avec les outils dont ils disposent. (Von Hippel, 2005).

Pour Chatelain, les modèles d'innovation pris en compte par les entreprises devraient – sur le modèle des *hackers* privilégiant la bidouille – tenir compte et privilégier les formes déviantes d'usage, afin d'innover sur un similaire à l'innovation par l'usage (Chatelain, 2006).

Pour Stewart et Gosain, les idéologies présentes dans le libre tendent à renforcer l'efficacité du développement du logiciel (Stewart et Gosain, 2006 (à paraître)).

Twidake et Nichols s'intéressent à l'utilisabilité de la gestion de bugs dans les logiciels libres. Leur recherche ethnographique sur un bugzilla (outils de gestion de bugs) révèle la pénurie d'expertise et l'importance du triage et de la classification des bugs, ce qui peut être fait grâce à des méta-données. Les auteurs soulignent l'importance de rendre explicite le contexte du bug, par exemple à l'aide de schémas, pour diminuer la complexité de la discussion visant à résoudre le problème (Twidale et Nichols, 2005).

Cohendet *et al.* étudient l'importance des communautés épistémiques et des communautés de pratique dans le cas Linux, et concluent que ces communautés sont le lieu réel de la création de connaissance tel que présenté sur la Figure 13, fonctionnant en « mode 2 » (Cohendet, Creplet et Dupouët, 2003).

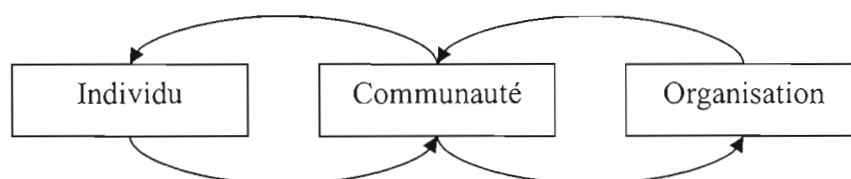


Figure 13 – Mécanisme de création et transfert de connaissance dans le logiciel libre

Lee et Cole opposent au modèle traditionnel de création de connaissance basé sur la firme un modèle de création de connaissance basé sur la communauté. Ils se fondent sur le processus de développement du noyau Linux, et dégagent quatre mécanismes : la connaissance fondatrice est largement partagée (grâce à la licence libre), une grande diversité de programmeurs est volontaire dans la communauté, l'effort de nombreux développeurs se fait sans contrôle hiérarchique, les contributions non planifiées deviennent des produits de haute qualité (Lee et Cole, 2003, p. 638). Les auteurs voient la critique par les pairs – rendue possible par la nature ouverte des

connaissances – comme une source de correction d’erreurs au service de la qualité du logiciel (p. 640). Ils voient aussi la vitesse de sortie de la version instable de Linux comme une source d’innovation et de création de connaissance (p. 641). Ils analysent la structure de Linux comme comportant un cœur avec un *leader* et une centaine de mainteneurs, et une périphérie composée de milliers de contributeurs, séparée en une équipe de développement et une équipe de report de bugs. Quand le nombre de contributeurs des parties périphériques croît, la création de connaissance augmente sans accentuer la complexité du projet, ce qui n’est pas le cas pour l’augmentation du nombre de contributeurs au cœur du projet (pp. 642-643). Le modèle de création de connaissance de la communauté de Linux fonctionne sur une plateforme internet de plusieurs à plusieurs et d’interactions directes entre les intervenants, ce qui fait la richesse de la connaissance partagée.

Hemetsberger et Reinhardt s'intéressent au problème de la transformation de la connaissance tacite sur internet. Pour les auteurs, la connaissance dans les communautés de pratique en ligne est partagée et co-crée à travers l'établissement d'un processus et de 'technologies' qui activent indirectement la connaissance (Hemetsberger et Reinhardt, 2004). Les auteurs montrent que l'activation de la ré-expérimentation, par exemple à travers la lecture d'une discussion ayant déjà eu lieu, (1) entraîne une diminution de la complexité et une mémoire de groupe transactive, (2) se fait par des conseils, de la franchise et de la participation légitime, (3) par une communication asynchrone et une expérimentation virtuelle. Ils cherchent comment les technologies d'internet sont utilisées pour adopter des processus d'apprentissage individuels et collectifs (Hemetsberger et Reinhardt, 2006 (à paraître)). En termes d'apprentissage, l'activation de la ré-expérimentation se fait en confrontation avec le code source et la mémoire de groupe transactive, par le contenu instructif et le discours. En termes de création de connaissance, l'activation de la ré-expérimentation se fait par pratique participative et par réflexion collective et expérimentation virtuelle. Leurs résultats sont en partie contradictoires avec ceux de Lee et Cole, qui

expliquent que l'apprentissage et le transfert de connaissances passent par la connaissance codifiée sous forme de code source, sans tenir compte des commentaires, des méthodes de gestion de conflit (qui comportent des résumés des modifications), etc. c'est-à-dire d'explication sur le code visant à réduire la complexité de sa compréhension. Au contraire, pour Hemetsberger et Reinhardt, l'apprentissage et le transfert de connaissance par le code à une composante importante enracinée dans les connaissances tacites des individus.

Ainsi, pour Hemetsberger et Reinhardt, l'innovation et la création de connaissances se font à travers l'usage des technologies internet en combinaison avec des pratiques interactives et discursives qui activent la ré-expérimentation.

Dans ce chapitre, nous avons présenté l'univers du logiciel, ses particularités économiques et l'aspect ayant trait à la propriété intellectuelle. Nous avons aussi présenté les aspects théoriques du logiciel libre, ainsi que différentes données permettant au lecteur de déchiffrer ce que représentent, dans les faits, les logiciels libres dans l'univers informatique. Nous avons terminé par une revue des travaux traitant de l'innovation et de la gestion des connaissances dans le logiciel libre. Dans le chapitre suivant, nous traiterons du cadre conceptuel délimitant notre recherche.

CHAPITRE III

CADRE CONCEPTUEL

Les chapitres I et II ont traité des principaux aspects liés aux domaines de la gestion des connaissances et du logiciel libre. À partir de ces deux champs, nous allons choisir et décrire dans ce chapitre le cadre conceptuel dans lequel va s'établir la recherche entreprise.

Rappelons notre problématique de recherche : nous étudions les conditions d'activation des communautés du logiciel libre engendrant de nouvelles connaissances dans un contexte d'écosystème dynamique.

3.1 Le logiciel libre

Nous avons eu l'occasion de voir dans le chapitre II différents aspects du logiciel libre. Comme nous avons eu l'occasion de le voir au cours de la revue de littérature, les valeurs suivies dans le logiciel libre sont tout à la fois importantes, diverses et – au moins en partie – en marge par rapport aux valeurs traditionnelles dans la société. Ainsi, il en va de l'importance de la contribution au sein d'un projet, non pas en argent, mais plus généralement (quoique pas uniquement) en terme de temps.

De même, au sein du logiciel libre, nous rencontrons des courants de pensée extrêmement divers allant de l'anarcho-communisme à l'anarcho-capitaliste, en passant par la théorie économique classique, et la théorie évolutionniste.

Tout ceci fait qu'une partie des personnes travaillant dans le logiciel libre se sentent politiquement impliquées, et certaines voient leurs travaux comme la matérialisation d'une idéologie (Demazière, Horn et Zune, 2006).

Cela fait du logiciel libre un terrain d'étude où il est important de prendre en compte les valeurs véhiculées par le projet.

De même que les valeurs du projet et des contributeurs peuvent être différentes et matière à débat important, les mécanismes de fonctionnement des différents logiciels sont différents, et ne forment pas un modèle dominant. Ainsi, Raymond dans son texte « la cathédrale et le bazar », attaque sous le vocable de cathédrale non pas des logiciels à code source fermé, mais des projets tels le projet GNU, qui bien qu'étant un logiciel libre ne favorise pas la diffusion de l'information. De plus, la structure de pouvoir dans les projets libres est assez variée, allant de l'autocratie à la méritocratie, avec des conceptions du mérite différentes. S'il ne s'agit pas d'une organisation structurée en tant que telle, il s'agit bien de mécanismes de fonctionnement communément partagés. Aussi, ces règles internes aux projets sont-elles importantes, et à prendre en considération.

Il est important de garder à l'esprit le fait que le logiciel libre est avant tout un logiciel, donc soumis aux caractéristiques informatiques (non excluabilité, non-rivalité, cumulativité) et au terrain qu'est l'ordinateur, c'est-à-dire une certaine relation avec la machine, comme nous avons été amenés à le voir dans la revue de littérature.

3.2 La gestion des connaissances

Nous avons choisi comme cadre de référence l'approche orientale de la gestion des connaissances pour son intégration de la création de connaissances. Ainsi, c'est à travers deux modèles que nous analyserons nos observations, à savoir le modèle SECI de Nonaka et Takeuchi (1994) et le modèle de la théorie de la firme créatrice de connaissance de Nonaka et Toyama (2005).

3.2.1 Le modèle SECI

Le modèle SECI (Socialisation, Extériorisation, Combinaison, Intériorisation) de Nonaka et Takeuchi (voir la section 1.3.2.2) est le modèle présentant la spirale de création de connaissance, tel présenté sur la Figure 14. Nous avons ajouté à ce modèle les étapes définies par Ebrahimi et Saives, permettant le passage d'une phase à une autre, à savoir : l'espace de partage pour la socialisation, l'artefact pour l'extériorisation, la redondance pour la combinaison et l'appropriation pour l'intériorisation.

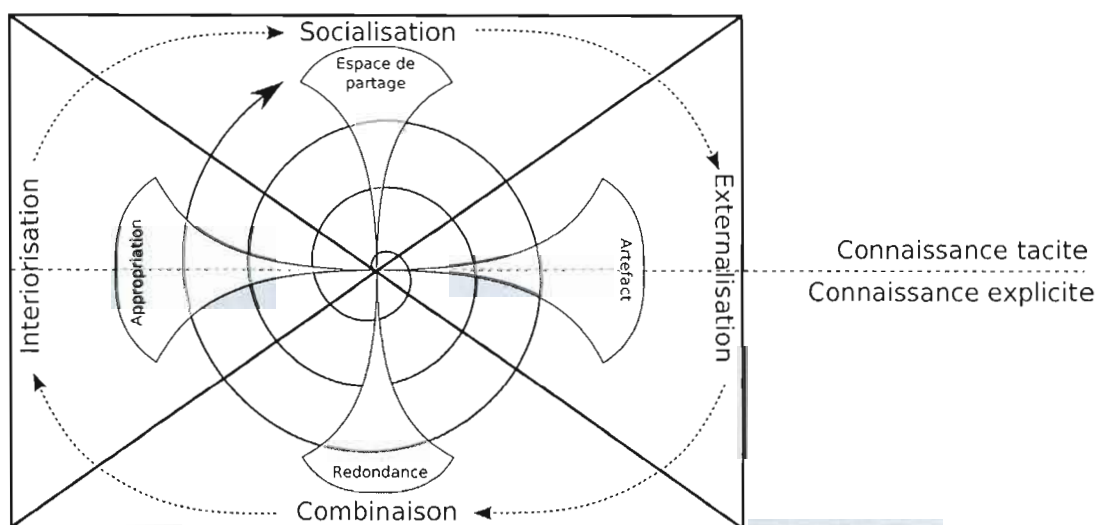


Figure 14 - Modèle SECI

3.2.2 Le modèle de théorie de la firme créatrice de connaissance

Nous avons choisi de baser notre approche de la gestion des connaissances à travers le modèle de Nonaka et Toyama (2005) de la théorie de la firme créatrice de connaissance (voir la section 1.4).

En effet, la théorie de la firme créatrice de connaissance ne voit pas la connaissance comme quelque chose d'absolu et d'infailible, contrairement aux visions

traditionnelles, mais quelque chose de toujours incomplet, en construction. Par ailleurs, ce modèle (voir Figure 15) donne un cadre dynamique à la connaissance au sein de l'organisation. Les frontières y sont en perpétuel mouvement. Cette approche nous paraissait particulièrement pertinente avec le milieu étudié.

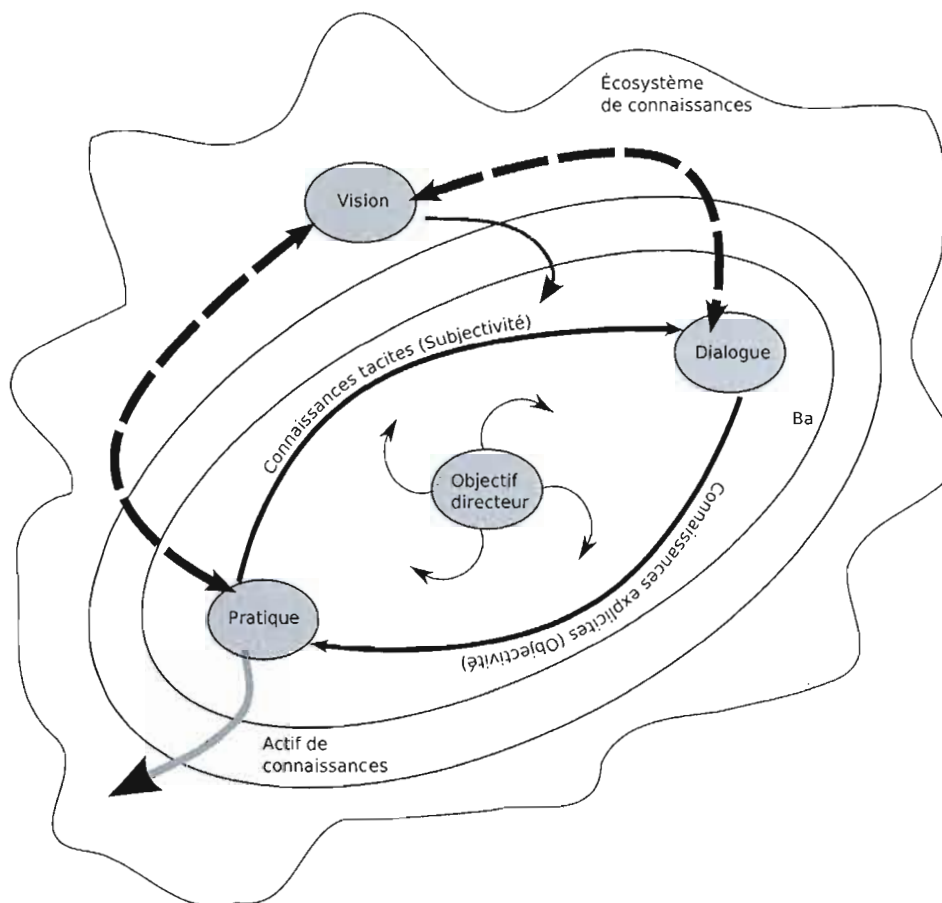


Figure 15 - Modèle de la firme créatrice de connaissance

3.3 Notre cadre conceptuel

Nous élaborons notre cadre conceptuel à partir de l'univers du logiciel libre (section 3.1) et de celui de la gestion des connaissances (section 3.2). Il s'agit d'analyser les valeurs, les croyances, les modes de fonctionnement du logiciel libre, à travers la

compréhension de la firme – au sens large – de la gestion des connaissances. Aussi nous présentons le cadre conceptuel tel qu'il apparaît sur la Figure 16.

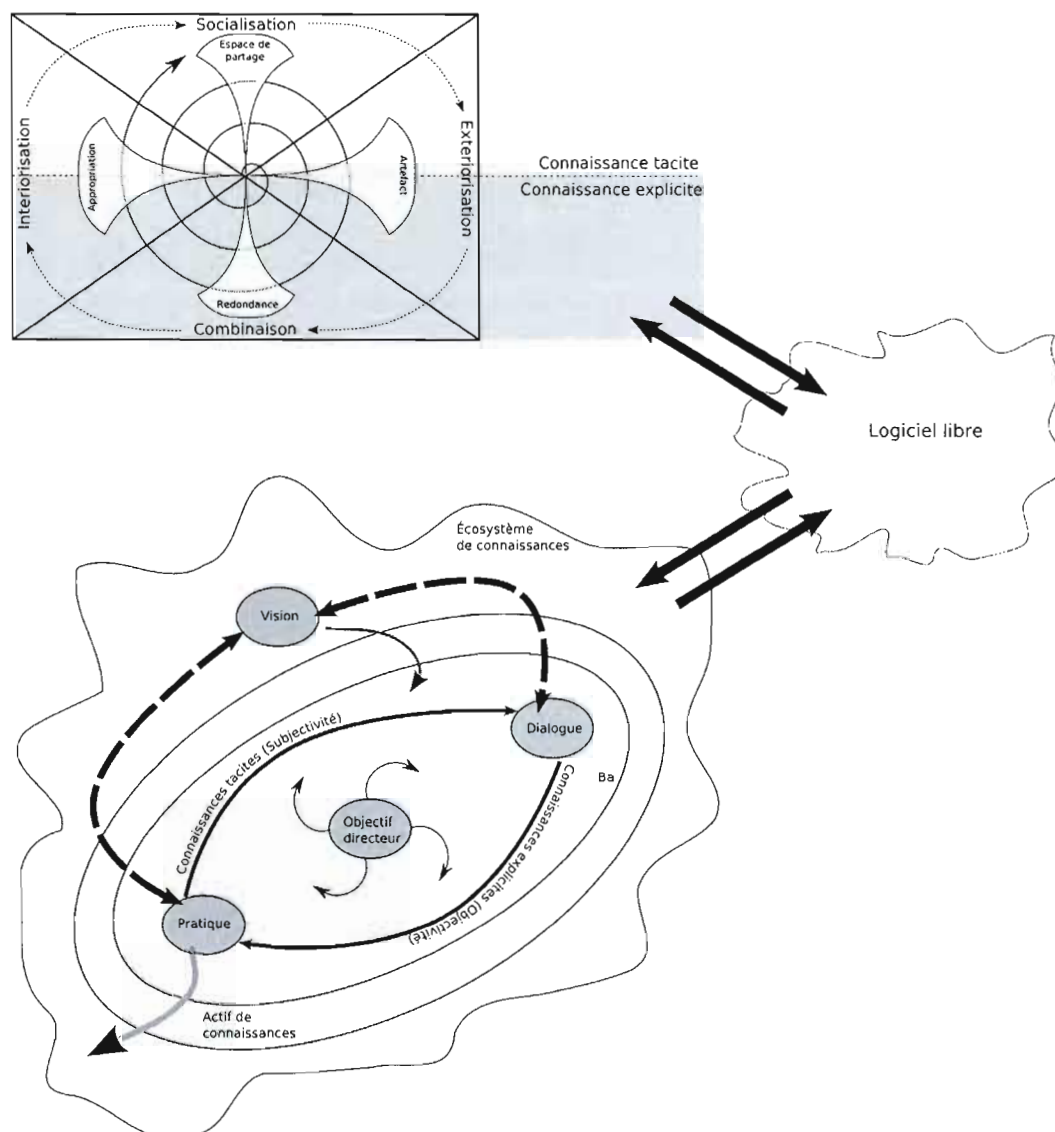


Figure 16 - Cadre conceptuel

Notre cadre conceptuel consiste à étudier et analyser la communauté du logiciel libre sous la double approche du modèle de la création de connaissance de Nonaka et Takeuchi (modèle SECI) et du modèle de la firme créatrice de connaissance.

Le modèle SECI nous fournit les outils permettant d'étudier, de comprendre et d'analyser la démarche créative se diffusant de l'individu à l'organisation, dans un processus dialectique dans la communauté. Le point de départ de l'analyse y est l'individu pour remonter vers l'aspect organisationnel.

Le modèle de la firme créatrice de connaissance offre un point de vue plus organisationnel, plus macroscopique des mécanismes à l'œuvre dans les communautés du libre. Le point de départ de l'analyse de ce modèle est l'organisation, et nous permet de descendre vers l'individu.

Ainsi, ces deux modèles complémentaires nous éclairent suivant deux approches différentes, mais avec un même objectif : comprendre les mécanismes de création de connaissance dans le logiciel libre.

Dans ce chapitre, nous avons présenté notre cadre conceptuel, délimitant la recherche du présent mémoire, qui se trouve à la croisée des chemins entre l'univers du logiciel libre et celui de la gestion des connaissances. Dans le chapitre à venir, nous allons présenter l'approche et la méthode utilisées pour obtenir nos résultats.

CHAPITRE IV

MÉTHODOLOGIE

Dans ce chapitre, nous traitons de l'aspect méthodologique de la recherche, nous accordant à des critères scientifiques rigoureux. Nous présenterons dans un premier temps l'approche de la recherche qui a été suivie, puis le choix méthodologique effectué, après quoi nous présenterons les lignes directrices de la démarche suivie, décrirons le choix du terrain et finalement le déroulement de la recherche.

4.1 L'approche de la recherche

Comme nous l'avons précédemment dit, cette recherche s'effectue selon un paradigme constructiviste. Pour Kuhn³² (Wikipedia, a) , un paradigme scientifique se définit comme :

- un ensemble d'observations et de faits avérés;
- un ensemble de questions en relation avec le sujet qui se posent et doivent être résolues;
- des indications méthodologiques (comment ces questions doivent être posées);
- comment les résultats de la recherche scientifique doivent être interprétés.

Le paradigme constructiviste est une conception de l'existence comme un « magma pré-établi de la connaissance, une conception où la connaissance émerge d'un processus humain continu de construction et de reconstruction. La connaissance émerge à l'intérieur d'une interaction entre les sujets; les objets et les projets » (Levy, 1994, p. 94). Ainsi, dans cette approche, le sens est enraciné à l'intérieur d'un

³² Kuhn, Thomas S., « Structure des révolutions scientifiques », 1962

« processus de manipulation, de configuration et de modification d'un monde incohérent qui aboutit à un processus infini de recherche de la signification de notre incorporation au monde ». Le chercheur constructiviste ne se reconnaît pas comme indépendant de son sujet de recherche – sans interactions, et neutre face à ce dernier. L'observateur y est un créateur de monde (p. 95), car il n'existe pas un monde, mais une multitude de perceptions de ce dernier, la perception faisant intégralement partie de ce monde. Aussi, le chercheur, en tant qu'acteur, crée son monde : *« le paradigme constructiviste est perçu comme un cadre conceptuel général non neutre et composé de théories et de valeurs qui influence directement notre vision du monde, nos construits et, bien sûr, nous-mêmes »*. (p. 96)

Au contraire, le paradigme réaliste ou positiviste voit le monde comme une entité indépendante de l'esprit, celui-ci étant objectivement existant, et rationnellement défini. Le monde y est prédéterminé et ordonné, il n'est pas attribuable à l'observateur neutre et pouvant par l'expérimentation déterminer les *« uniques sources de "vérité" »* (p. 94), celles-ci étant intrinsèques au monde. Levy simplifie ces différences entre positivisme et constructivisme en énonçant que *« pour le réalisme, la certitude est routine alors que pour le constructivisme, le doute est routine »* (p. 95).

Le choix d'un paradigme doit se faire selon les valeurs et croyances du chercheur. Nous partageons l'avis de l'auteur, que *« même si les méthodes d'investigation interprétatives peuvent être inexactes sur le plan statistique ou mathématique, elles sont toutefois porteuses de formes de connaissances plus complètes et plus vastes »* (p. 99). C'est aussi ce qu'entend Merton³³ lorsqu'il présente l'opposition entre les chercheurs constructivistes qui pourraient dire *« nous n'affirmons pas que ce que*

³³ Robert K. Merton, « éléments de théorie et de méthode sociologique », Plon, Paris, 1965, p.334-335 d'après Breton et Proulx

nous avançons soit la vérité, mais au moins c'est significatif» contrairement aux chercheurs positivistes annonçant que « *nous ne savons pas si ce que nous avançons est significatif mais au moins c'est vrai* » (Breton et Proulx, 1989).

Les paradigmes réaliste et constructiviste ont tous deux leurs intérêts, toutefois, ils sont incommensurables, donc non comparables (Levy, 1994, p. 97). Ainsi, si la validité et la fiabilité sont des facteurs de crédibilité dans le cadre d'une recherche dans un cadre positiviste, elles n'ont pas de sens – en tant que tel – dans une recherche constructiviste. C'est une des raisons pour lesquelles le constructivisme se distingue par son acceptation d'une multiplicité de résultats. Cela ne veut toutefois pas dire que les résultats ne sont pas soumis à une certaine rigueur de construction et ne doivent pas être comparés avec la réalité, dans le but de rester en adéquation avec cette dernière.

4.2 Le choix méthodologique

Le choix d'une méthodologie découle de l'approche paradigmatique suivie, et l'approche la plus adaptée au paradigme constructiviste est l'approche qualitative (Levy, 1994, p. 93), mieux à même de remplir l'objectif principal de cette approche, c'est-à-dire « *de construire des représentations des perceptions du monde vécu et, ainsi, de chercher à diverger vers des conceptions multiples et émergentes de l'expérience et des phénomènes* » (p. 94).

Aussi, dans le cadre d'une recherche d'ordre qualitative, nous avons choisi une approche découlant de la méthode de la théorisation ancrée. Cette méthode est avant tout une démarche de théorisation (Paillé, 1994, p. 149), c'est-à-dire une démarche visant à dégager le sens d'événements, à lier dans un schéma explicatif divers éléments d'une situation, à renouveler la compréhension d'un phénomène en le mettant différemment en lumière. Plus que le résultat, c'est le processus qui est

important. Le résultat de cette théorie doit être solidement ancré dans les données empiriques recueillies. Ainsi,

À la différence des théories traditionnelles, d'abord créées, ensuite vérifiées (validées), la plupart par des chercheurs différents et dans des conditions différentes, une théorie ancrée est construite et validée simultanément par la comparaison constante entre réalité observée et l'analyse émergente (p. 150).

Pour l'auteur, « *il s'agit d'une démarche itérative de théorisation progressive d'un phénomène* » (p. 151).

L'une des caractéristiques majeures de l'analyse par la théorisation ancrée est la simultanéité de la collecte et de l'analyse. « *Le chercheur ne se situe pas tant dans une démarche de codification d'un corpus existant [...] que dans un processus de questionnement* ».

Le corpus de données pouvant être utilisé comme étant varié et divers. En effet, « *l'analyse par théorisation ancrée est une méthode d'analyse de données empiriques, quel que soit, en fin de compte, le support du corpus de base* » (p. 152). Le choix de sites et d'informateurs y est conditionné par l'analyse en évolution plutôt que prédéterminé sur la base de critères « objectifs ». De plus, les instruments utilisés demeurent provisoires, et ne nécessitent pas d'être toujours identique à titre de comparaison, contrairement à ce qui serait nécessaire dans le cadre d'une recherche positiviste (p. 153). Aussi, le corpus de données peut être fort varié : « *notes de terrains, transcriptions d'entrevue formelles ou informelles, documents variés, etc.* » (p. 151).

4.3 La démarche méthodologique

Paillé présente les étapes suivantes comme reflétant « *le processus réel de théorisation d'un corpus, mais il ne faudrait pas comprendre qu'elles sont exclusives les unes des autres et que la progression est linéaire* » (pp. 153-154) :

1. la codification initiale : il s'agit du point de départ de l'analyse par théorisation ancrée dont le but est de répondre aux questions « *qu'est-ce qu'il y a ici? Qu'est-ce que c'est? De quoi est-t-il question?* » (p. 154) à l'aide d'un codage de données s'apparentant à des métadonnées (des données sur des données). Il faut faire ressortir ici les propriétés essentielles, il n'est pas possible de faire ressortir dans la codification tout ce que traduit la donnée analysée. Le chercheur doit trouver un compromis entre la trop grande prudence et l'interprétation à outrance, afin de rester fidèle aux données;
2. le passage de la codification à la catégorisation : il s'agit d'attribuer aux codes de l'étape 1 différents types : les *codes conceptuels*, relevant de concepts, les *codes-chevilles* relevant d'un ensemble de codes regroupés dans une idée plus large, de *codes centraux* indiquant des idées intéressantes, de *codes de réserves* semblant ne mener nulle part, de *codes socio-démographiques* sur les personnes et les lieux (pp. 157-159);
3. la catégorisation : il s'agit de porter l'analyse à un niveau conceptuel, en enrichissant et en englobant les phénomènes se dégageant des données. Pour ce faire, le chercheur doit repasser à travers les données non codées et leurs donner des catégories, en restant au niveau conceptuel. Les questions à se poser à cette étape sont : « *Qu'est-ce qui se passe ici? De quoi s'agit-il? Je suis en face de quel phénomène?* » (p. 159). Le chercheur rend donc ici visibles les phénomènes d'importance en les mettant en perspective. Au terme des étapes 1 et 3, le chercheur dispose de deux lectures différentes, permettant d'avoir les événements (codification) et leur contexte (catégorisation);
4. la construction et la consolidation des catégories : il s'agit du travail effectué sur les catégories, afin de les clarifier, détailler, préciser, corriger, remplacer, fusionner ou diviser. Pour se faire, le chercheur (1) définit les catégories,

(2) dégage leurs propriétés, c'est-à-dire cherche à déterminer leurs compositions, leurs caractéristiques et leurs attributs, (3) spécifie les conditions sociales qui fondent la formulation d'une catégorie, c'est-à-dire ce qui doit être présent pour que la catégorie s'applique et (4) identifier les diverses formes de la catégorie;

5. la mise en relation : cette étape met en relation les différentes catégories pour les structurer dans une hiérarchie, sous forme de graphe accompagné de mémos d'explications. Cela peut se passer par une approche empirique (à partir du terrain), une approche spéculative (faire appel à la logique et l'expérience du chercheur) ou une approche théorique (faire appel aux travaux existant dans le domaine);
6. l'intégration : il s'agit, en fonction de ce vers quoi le chercheur a été amené à progresser (dans une approche par théorisation ancrée, l'analyse peut porter le chercheur dans une autre direction que celle de départ), de redonner un sens (comprendre une direction claire) au travail, afin de ne pas perdre le lecteur. Cela se fait en gardant à l'esprit la question « *quel est le problème principal? Sur quoi porte finalement l'étude?* » (p. 172). La réponse à ces questions permet de délimiter l'objet précis que deviendra l'analyse;
7. la modélisation : elle consiste « *à reproduire le plus fidèlement possible l'organisation des relations structurelles et fonctionnelles caractérisant un phénomène, un événement, un système, etc.* » (p. 174). Ainsi, il y faut dégager les caractéristiques importantes, d'un phénomène, son déroulement, ses conséquences, etc. Il s'agit d'une reconstitution dynamique du phénomène observé;
8. la théorisation : c'est la dernière étape, composée de (1) l'échantillonnage théorique, consistant à donner un exemple illustratif de ce qu'il se passe.

Ainsi, l'échantillonnage s'effectue tout au long de la recherche. Au début l'échantillonnage vise à cerner les variations, puis, il permet l'enracinement de la théorie et son raffinement, (2) la vérification des implications théoriques du modèle, c'est-à-dire « *indiquer le plus précisément possible quelles implications découlent logiquement de la théorie ou des explications alternatives, pour ensuite vérifier dans le corpus ou lors d'observations ou d'entrevues subséquentes si ces implications se manifestent effectivement* » (pp. 178-179). Cela permet de vérifier la fiabilité de la théorie. Enfin (3) l'induction analytique, qui est une stratégie globale d'analyse inductive, visant à valider la théorie par la recherche de « cas négatif ».

4.4 Le choix du terrain

Comme terrain d'étude, il nous est apparu important de trouver un projet libre particulièrement innovant en terme de création de connaissance, ce qui nécessitait d'étudier un projet d'une taille assez importante et évoluant assez rapidement.

Par ailleurs, dans une écologie du libre changeante, il nous paraissait intéressant d'étudier un projet tentant de rentrer en concordance avec le profil de nouveaux utilisateurs, tout en restant dans la culture traditionnelle du logiciel libre. Aussi, le choix devait se porter sur un projet relativement récent ou extrêmement dynamique dans ses mécanismes internes de fonctionnement.

Il nous paraissait aussi important d'étudier un projet étant amené à travailler en collaboration avec de nombreux autres projets, ce qui plaçait – entre autres choix possibles – l'index sur une distribution.

Enfin, la présence d'une réunion internationale à Montréal des contributeurs de la distribution Ubuntu, distribution récente, fortement créatrice en termes de connaissance, et ayant pour objectif de bien s'adapter aux nouvelles contraintes de

l'écologie du libre a fait de ce choix une résolution judicieuse et un terrain d'étude particulièrement intéressant.

Toutefois, si la communauté Ubuntu est le cœur, le point de départ et le point d'arrivée de la recherche, nous ne nous limiterons pas à cette communauté, cette dernière étant amenée à travailler avec d'autres communautés.

Il nous paraissait important de ne pas se limiter à cette communauté, car en la segmentant artificiellement de la sorte, nous risquons tel que le décrit Laperriere avec des « *situations et populations ainsi découpées à des fins de contrôle et de comparaison [de s'éloigner] progressivement des situations naturelles, mettant en jeu la pertinence des données* » (Laperriere, 1997, p. 336), or comme le dit Gusdorf « *l'espace de la connaissance constitue un domaine unitaire* » (Gusdorf, 1974, p. X). C'était nous semble-t-il a priori et a posteriori, une condition sine qua non de significativité de la recherche.

4.5 Le déroulement de la recherche et description du terrain

Cette recherche s'est déroulée entre les mois d'octobre 2005 et d'août 2006. Nous sommes par ailleurs confrontés au logiciel libre depuis plus de dix ans, et à Ubuntu depuis la version 5.04 (avril 2005).

Au cours de la recherche, nous avons été amenés à participer à la réunion de contributeurs d'octobre 2005 à Montréal (nommée « UBZ » pour *Ubuntu Below Zero*) en tant qu'observateur. Cette réunion était ouverte à tous.

Nous avons aussi utilisé de façon importante les moyens de communication qu'utilise la communauté Ubuntu, c'est à dire l'IRC, les *mailing-lists* (principalement ubuntu-dev et ubuntu-dev-announce), le web, les forums, etc.

Pour notre recherche, nous avons principalement été amenés à travailler avec ces *mailing-lists* (la liste des différentes listes existantes est présentée en Annexe 1) et les groupes de discussions sur IRC, ce qui s'apparente à une observation non-participante, puisqu'il s'agit des lieux d'échanges de la communauté, où les différents individus communiquent les uns avec les autres, débattent, discutent. Il s'agit en quelque sorte d'un agora public d'échanges entre personnes, les échanges étant enregistrés et consultables ultérieurement, ce que nous avons eu l'occasion de faire, « rentrant » pour ainsi dire dans les conversations des contributeurs, qu'elles soient présentes ou passées.

Lorsque les décisions faisaient référence à d'autres projets, nous avons « suivi » le projet, en utilisant tout le corpus de données, à savoir généralement les sites web, les archives des *mailing-lists*, afin de replacer le contexte de la problématique.

Nous avons aussi été amenés à effectuer des entrevues informelles, non directives, avec divers contributeurs, différemment impliqués au sein du projet Ubuntu. Certains de ces entretiens ont été menés durant les deux semaines d'« UBZ » directement avec les personnes, d'autres entretiens ont été effectués par messagerie instantanée (couramment utilisé dans la communauté) ou encore par courriel.

Dans ce chapitre, nous avons présenté l'approche constructiviste de la recherche, et le choix méthodologique de la théorisation ancrée en découlant. Nous avons décrit cette démarche, ainsi que les facteurs ayant conduit au choix du terrain.

À partir des nombreuses données, nous avons procédé suivant la méthode précédemment décrite section 4.3, c'est-à-dire que nous avons fait ressortir les catégories émergentes des données, puis nous les avons regroupées et hiérarchisées pour aboutir à la catégorisation présentée sur la Figure 17, et sur laquelle nous reviendrons en détail dans le chapitre 6.

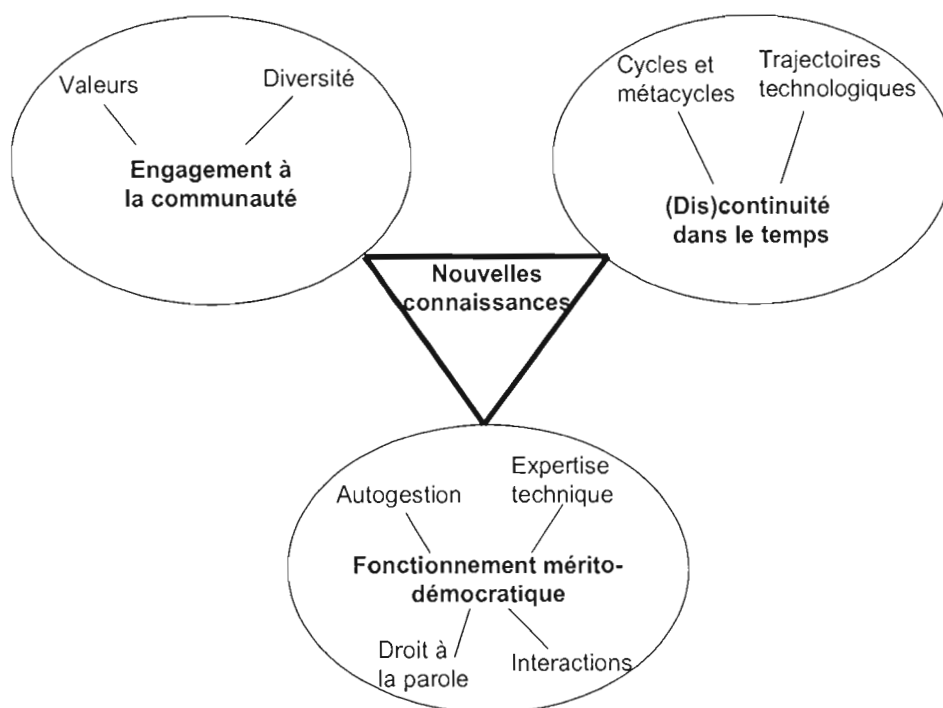


Figure 17 - Catégories et sous catégories émergentes de la recherche

L'analyse au chapitre 7 correspond à une mise en relation avec le cadre conceptuel développé au chapitre 3, afin de faire ressortir les éléments déterminants concernant un modèle de création de connaissance adapté au libre ainsi que les conditions d'activation des communautés des connaissances.

Dans le chapitre 5 à venir, nous présenterons au lecteur les principaux éléments du projet Ubuntu et du monde du logiciel libre.

CHAPITRE V

PRÉSENTATION DE L'UNIVERS D'UBUNTU ET DU LOGICIEL LIBRE

Dans ce chapitre, nous décrirons l'univers de la communauté Ubuntu et plus généralement du logiciel libre dans lesquels l'étude a été menée. Nous sommes conscients que l'explication du fonctionnement d'un système d'exploitation dépasse de très loin les propos de ce mémoire, mais nous serons toutefois amenés à décrire des dispositifs techniques, qui ont une importance de par ce qu'ils permettent de faire en terme d'organisation. Notons bien que le propos technique n'est pas en soi notre sujet ici. Nous l'aborderons donc volontairement d'un point de vue simplifié, afin de faire ressortir les enjeux pour le non-spécialiste.

Dans une première partie, nous allons tenter d'esquisser le fonctionnement d'une distribution du type Ubuntu et de ce qu'il représente d'un point de vue technique. Dans une deuxième partie, nous traiterons de son fonctionnement, des valeurs véhiculées, des moyens de communication utilisés, de liens avec d'autres projets externes et de quelques projets internes. Nous dresserons alors un schéma récapitulant l'écosystème d'Ubuntu.

5.1 GNU, Linux, Distributions et projets importants

Le logiciel libre est caractérisé par une multitude de projets et de sous projets interconnectés – ou non – et de différentes tailles. Dans un système d'exploitation, il existe différentes couches, certaines obligatoires (noyau, pilotes), d'autres nécessaires

à un système d'exploitation moderne (système de fenêtrage³⁴), enfin d'autres optionnelles (applications). Ainsi, Linux (section 5.1.1) est le noyau, Gnome (section 5.1.3) et KDE (section 5.1.4) sont des environnements utilisateur.

5.1.1 Le système GNU/Linux

Le système GNU/Linux est l'association du noyau Linux avec les outils GNU.

Linux – ou **GNU/Linux** – est un système d'exploitation libre, multitâche, multi-plateforme et multi-utilisateur de type Unix.

Il tire son nom d'une de ses parties, à savoir de son noyau, initié par Linus Torvalds en 1991. Il s'agit d'un composant central et de bas niveau qui s'occupe de fournir aux logiciels une interface pour communiquer entre eux et avec le matériel.

Le projet GNU (L'acronyme récursif GNU signifie GNU is Not Unix), dont l'objectif est de fournir un système d'exploitation complet sous licence libre, a quant à lui été initié par Richard Stallman en 1984 et est soutenu principalement par l'association FSF (Free Software Foundation)³⁵..

UNIX est le nom d'un système d'exploitation créé en 1969, à usage principalement professionnel, conceptuellement ouvert et fondé sur une approche par laquelle il offre de nombreux petits outils chacun doté d'une mission spécifique, multitâche et multiutilisateur. Il a donné naissance à une famille de systèmes, dont les plus populaires en 2005 sont GNU/Linux, *BSD et Mac OS X³⁶.

5.1.2 Les distributions

Nous donnerons la définition suivante d'une distribution : elles « *rassemblent les composants d'un système GNU/Linux dans un ensemble cohérent et stable facilitant*

³⁴ Un système de fenêtrage permet d'utiliser un environnement graphique, avec une souris et des fenêtres. Sous Unix, le système de fenêtrage le plus fréquent est le logiciel X.

³⁵ Source : <http://www.gnu.org/gnu/thegnuproject.fr.html>

³⁶ Source : <http://fr.wikipedia.org/wiki/Unix>

son installation, utilisation et maintenance. Elles comprennent donc le plus souvent un logiciel d'installation et des outils de configuration »³⁷. Sur le site internet du projet Debian (défini en section 5.2.5), on trouve la métaphore suivante : une distribution, « c'est un peu comme une tour. À la base il y a le noyau. Au-dessus les utilitaires fondamentaux. Puis ce sont tous les logiciels que vous lancez sur l'ordinateur. Au sommet de la tour, il y a [la distribution] — qui organise soigneusement l'ensemble afin que tout puisse fonctionner correctement »³⁸. C'est ceci qui est représenté sur la Figure 18.

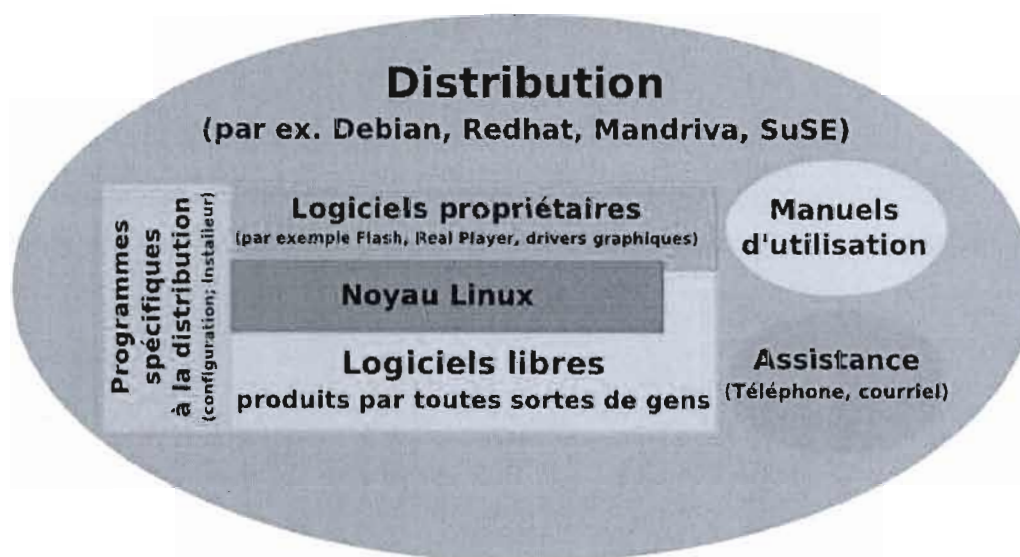


Figure 18 – Architecture d'une distribution³⁹

³⁷ Source : http://fr.wikipedia.org/wiki/Distribution_Linux

³⁸ Source : <http://www.us.debian.org/intro/about>

³⁹ Source : http://fr.wikipedia.org/wiki/Distribution_Linux

5.1.3 GNOME

GNOME est l'environnement graphique utilisé dans le cadre d'Ubuntu (voir copie d'écran en annexe 4). GNOME s'appuie sur les couches de base en dessous de lui, c'est-à-dire le noyau, diverses bibliothèques, le système de fenêtrage.

Le projet GNOME a pour but de créer un environnement utilisateur complet, libre et facile d'utilisation, ainsi qu'une plate-forme de conception logiciel solide pour les développeurs d'applications. [...] GNOME est ...

- **Libre.** GNOME est un Logiciel Libre et fait partie du projet GNU, il vise à offrir à ses utilisateurs et développeurs un contrôle total de leur environnement, de leurs applications et de leurs données. [...]
- **Ergonomique.** Pour que l'utilisation de GNOME soit une expérience agréable s'améliorant en permanence, un groupe de travail se concentre sur l'ergonomie.
- **Accessible.** Le but est de permettre aux personnes souffrant d'un handicap de participer aux activités essentielles de la vie, ce qui comprend le travail et l'utilisation des services, des produits et de l'information. [...]
- **International.** GNOME est d'ores et déjà utilisé, conçu et documenté dans des dizaines de langues. Notre but est de permettre la traduction de tout logiciel GNOME dans n'importe quelle langue. [...]
- **Organisé.** GNOME se veut une communauté organisée, avec à sa base plusieurs centaines de membres, un conseil élu et des groupes responsables de l'ergonomie, de l'accessibilité et de la qualité du logiciel. Le cadre de chaque nouvelle version de GNOME est défini par les membres du groupe de lancement de GNOME à un rythme de deux versions par an.
- **Reconnu.** Le projet GNOME est reconnu pour sa valeur bien au-delà de la communauté internationale qui le fait vivre. Il reçoit l'appui des principales sociétés investies dans Linux et Unix comme HP, Mandrake, Novell, Red Hat, et Sun. [...]
- **Une communauté.** Plus que toute autre chose encore, GNOME est une communauté de volontaires qui travaillent à la conception, à la traduction et au maintien de la qualité du logiciel qu'ils développent ensemble le plus souvent dans la bonne humeur⁴⁰.

⁴⁰ Source : <http://www.gnomefr.org/apropos/>

Le projet GNOME possède ainsi une HIG (pour *Human Interface Guide*) très stricte afin de simplifier au maximum l'utilisation pour l'utilisateur.

5.1.4 KDE

KDE, comme GNOME est un environnement graphique utilisable sous X⁴¹ (voir copie d'écran en annexe 4).

KDE is a network transparent contemporary desktop environment for UNIX workstations. KDE seeks to fill the need for an easy to use desktop for Unix workstations, similar to the desktop environments found under the MacOS or Microsoft Windows. We believe that the UNIX operating system is the best operating system available today. In fact UNIX has been the undisputed choice of the information technology professional for many years. When it comes to stability, scalability and openness there is no competition to UNIX. However, the lack of an easy to use contemporary desktop environment for UNIX has prevented UNIX from finding its way onto the desktops of the typical computer user in offices and homes. UNIX dominates the server market and is the preferred computing platform for computing professional and scientists. Without UNIX the internet would not be. But UNIX did not address the needs of the average computer user. This fact is particularly unfortunate since a number of implementations of UNIX (Debian GNU/Linux, FreeBSD, NetBSD etc.) are freely available on the internet. All of which are of exceptional quality and stability⁴².

Toutefois, l'emphasis n'est pas mise sur la simplicité d'utilisation, mais sur les fonctionnalités offertes à l'utilisateur.

Bien que les applications développées pour GNOME soient utilisables pour KDE, et inversement, la vision est différente entre ces communautés, et cela génère sur les forums, *mailing-lists*, ... des messages à caractère tendancieux – qualifié de troll⁴³ –

⁴¹ X est le système de fenêtrage généralement utilisé sous GNU/Linux.

⁴² Source : <http://kde.org/whatiskde/>

⁴³ Sur wikipédia, le troll est qualifié de « *message dont le caractère est susceptible de générer des polémiques ou étant excessivement provocateur, sans chercher à être constructif* ». Source : [http://fr.wikipedia.org/wiki/Troll_\(Internet\)](http://fr.wikipedia.org/wiki/Troll_(Internet))

donnant lieu à une guerre sans fin et souvent une avalanche de messages. Ainsi, Linus Torvalds a récemment déclaré :

I personally just encourage people to switch to KDE.
This "users are idiots, and are confused by functionality" mentality of Gnome is a disease. If you think your users are idiots, only idiots will use it. I don't use Gnome, because in striving to be simple, it has long since reached the point where it simply doesn't do what I need it to do. Please, just tell people to use KDE⁴⁴.

Ubuntu utilise l'environnement graphique GNOME, toutefois, une version d'Ubuntu nommée Kubuntu est l'équivalent utilisant l'environnement graphique KDE.

Ainsi, nous pouvons schématiser assez grossièrement le fonctionnement d'un système informatique de type GNU/Linux tel que présenté sur la Figure 19.

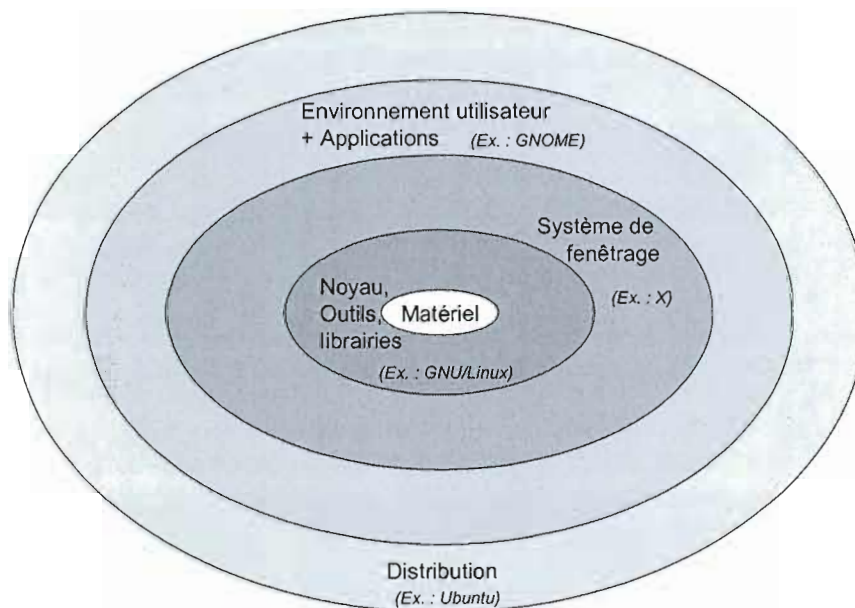


Figure 19 - Schéma simplifié de l'architecture matériel / logiciel

⁴⁴ Source : <http://mail.gnome.org/archives/usability/2005-December/msg00021.html>

5.2 Le projet Ubuntu

5.2.1 Description

Ubuntu est une des distributions Linux les plus récentes, mais aussi les plus en vogue depuis la sortie de la version Warty en 2004. La description sur le site internet de la distribution dit ceci :

Ubuntu est une distribution Linux qui réunit stabilité et convivialité. Elle s'adresse aussi bien aux particuliers qu'aux professionnels, débutants ou confirmés qui souhaitent disposer d'un système d'exploitation libre et sécurisé.

'Ubuntu' est un ancien mot africain qui signifie 'humanité aux autres'. Ubuntu signifie également 'Je suis ce que je suis grâce à ce que nous sommes tous'. La distribution Ubuntu Linux apporte l'esprit Ubuntu au monde logiciel⁴⁵.

Ubuntu est une distribution libre, qui souhaite le rester. Ainsi, Mark Shuttleworth déclare qu'Ubuntu « *est, et sera toujours entièrement LIBRE et gratuit, c'est donc un parfait terrain de jeu pour ceux qui veulent construire quelque chose et innover* »⁴⁶.

La vocation première d'Ubuntu pour Mark Shuttleworth⁴⁷, est de corriger le « bug #1 », qui est décrit ainsi :

Microsoft has a majority market share in the new desktop PC marketplace. **This is a bug, which Ubuntu is designed to fix.**

Microsoft has a majority market share | Non-free software is holding back innovation in the IT industry, restricting access to IT to a small part of the world's population and limiting the ability of software developers to reach their full potential, globally. This bug is widely evident in the PC industry.

Steps to repeat:

1. Visit a local PC store.

What happens:

2. Observe that a majority of PC's for sale have non-free software pre-installed

⁴⁵ Source : <http://www.ubuntu-fr.org/>

⁴⁶ Source : Interview de Mark Shuttleworth par Slashdot, traduit par Framasoft sur <http://www.framasoft.net/article3752.html>

⁴⁷ Source : FAQ de Mark Shuttleworth traduite par Framasoft : <http://www.framasoft.net/article624.html>

3. Observe very few PC's with Ubuntu and free software pre-installed

What should happen:

1. A majority of the PC's for sale should include only free software like Ubuntu
2. Ubuntu should be marketed in a way such that its amazing features and benefits would be apparent and known by all.
3. The system shall become more and more user friendly as time passes⁴⁸.

Le développement d'Ubuntu a commencé en 2004. Depuis la première version Warty, le rythme de sortie est d'une version tous les six mois et ce rythme restera constant, à moins qu'il en soit décidé autrement par la communauté. Notons toutefois que la sortie de la version Dapper a eu un retard de six semaines, l'objectif pour cette version était de faire en sorte qu'elle soit le plus stable possible afin d'être utilisée en entreprise. De plus, cette version bénéficie d'un support long terme (trois ans pour les versions Desktop, cinq ans pour les versions serveur). Les différentes versions sont :

- Warty Warthog (Le Phacochère Verruqueux) 4.10⁴⁹;
- Hoary Hedgehog (Le Hérisson Vénérable) 5.04;
- Breezy Badger (Le Blaireau Jovial) 5.10;
- Dapper Drake (Le Canard Pimpant) 6.06;
- Edgy Eft (Jeune Salamandre Énervée) 6.10 (prochaine version)⁵⁰.

Les auteurs de la distribution la distinguent, par rapport aux autres distributions concurrentes de la sorte :

Basée sur Debian, l'une des distributions les plus reconnues, les plus technologiquement avancées et des mieux supportées, Ubuntu vise à créer une distribution qui fournit un système Linux à jour et cohérent pour les machines de bureau et les serveurs. Ubuntu inclut de nombreux paquets rigoureusement choisis et issus de la distribution Debian, et conserve aussi son puissant gestionnaire de paquets qui permet l'installation facile de programmes et leur

⁴⁸ Source : Launchpad, Malone. L'emphasis est de nous.

⁴⁹ Le numéro de version 4.10 correspond à [l'année de sortie (ici 2004)].[le mois de sortie (ici octobre)].

⁵⁰ Source : <http://doc.ubuntu-fr.org/versions>

suppression de façon propre. Contrairement à la plupart des autres distributions fournies avec de nombreux logiciels plus ou moins utiles, la liste des paquets d'Ubuntu est réduite à un nombre d'applications de haute qualité⁵¹.

5.2.2 Les valeurs et le fonctionnement

La distribution Ubuntu a adopté ses propres façons de faire et ses valeurs, basées sur le libre. Ainsi, un code de conduite (disponible en annexe 2) est à signer pour être considéré comme « *ubunteros* », qualificatif identifiant les personnes s'impliquant dans la communauté Ubuntu.

La gouvernance d'Ubuntu est assurée par le quintuplet présenté en Figure 20.

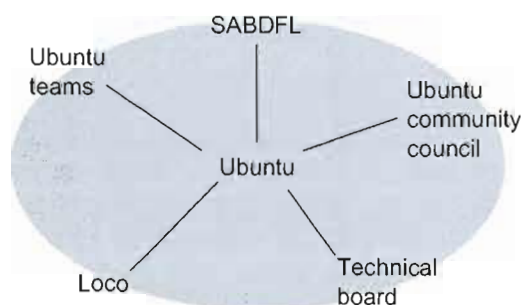


Figure 20 - Gouvernance d'Ubuntu

- SABDFL est le pseudonyme de Mark Shuttleworth, et signifie « *Self-Appointed Benevolent Dictator for Life* ». Il s'agit de l'initiateur d'Ubuntu, et le principal sponsor financier permettant de faire fonctionner la compagnie Canonical Ltd (voir section 5.2.4). Il participe activement à la vie de la distribution;
- Ubuntu teams correspondent à des équipes de personnes spécialisées autour d'un thème ou d'un logiciel spécifique. Ils prennent alors part aux décisions

⁵¹ Source :

<https://help.ubuntu.com/ubuntu/desktopguide/fr/about-ubuntu.html#ubuntu-difference>

les concernant. Ainsi, il existe des équipes s'occupant du noyau Linux, de l'esthétique de la distribution, ...;

- Loco, pour *local community*, ou communautés locales. Il s'agit de communautés regroupées autour d'une spécificité, comme la communauté francophone (communauté linguistique), ou la communauté canadienne (communauté géographique);
- Le *technical board* est un groupe d'individus s'occupant spécifiquement des aspects techniques de la distribution, d'une façon générale. Ils sont reconnus pour leurs connaissances techniques et leur appartenance à la communauté;
- *Ubuntu community council* : conseil représentant les intérêts de la communauté. Ses membres sont des personnes s'impliquant de différentes manières dans la communauté (participation technique ou non).

5.2.3 Les moyens de communication

- The Fridge : site web d'informations concernant la distribution, s'y trouvent des *newsletters*, des comptes-rendus de réunion, ...;
- Ubuntu Weekly Newsletter : journal d'information disponible sur le wiki d'Ubuntu, traitant de sujet divers, paraissant toutes les semaines, ayant pour thème Ubuntu;
- Planet Ubuntu : agrégateur de blogs, permet de récupérer à partir de différents blogs de développeurs et utilisateurs d'Ubuntu des billets concernant la distribution ou son développement, ou des anecdotes s'y rapportant;
- Forums : lieu d'échange, principalement de la communauté, où les utilisateurs s'entraident;
- Sites Web traditionnels : délivre l'information institutionnelle standard, sur le projet, son but, ses valeurs, son fonctionnement, ...;

- *Mailing-lists* : lieu d'échange des développeurs et / ou utilisateurs avancés / intéressés à suivre l'avancement d'un projet (dans le cas du logiciel libre, la distinction entre développeurs et utilisateurs n'est pas nette);
- IRC : Lieu virtuel de communication instantanée, autour de thématiques précises (un logiciel par exemple), ou plus générales (la distribution Ubuntu). Il existe différents canaux de conversation ayant des fins différentes (il existe par exemple un canal servant aux différentes réunions ayant lieu concernant la distribution);
- Wiki : site internet où l'utilisateur peut modifier la page, devenant alors corédacteur avec les autres utilisateurs. Dans le cas d'Ubuntu, le Wiki est utilisé afin de créer la documentation sur la distribution, de tenir les utilisateurs informés des prochaines réunions et ordres du jour s'y rapportant, d'annoncer d'éventuelles modifications pour un logiciel;
- Launchpad : Site internet central au développement, créé pour Ubuntu (mais aussi utilisé dans le cadre d'autres projets libres), regroupant le traqueur de bugs (permet aux utilisateurs de signaler des bugs dans les logiciels, de faire des demandes spécifiques, ...), centralisant les traductions, et permettant aux utilisateurs de traduire simplement les logiciels, de traquer des fonctionnalités en développement à travers des spécifications et de leur accorder des statuts (urgences, acceptation par les développeurs, ...) et de faire le lien avec d'autres pages de description de modification (Wiki).

Ces différents médiums de communication offrent un large éventail, suivant les avantages / inconvénients de chacun, et permettent de créer de nombreux lieux d'information, et de débat.

De plus, lors de chaque sortie de distribution, et à mi-parcours entre deux distributions – c'est-à-dire tous les trois mois – une réunion nommée « *sprint* » ayant pour thème le développement de la version courante de la distribution a lieu. Nous étions présents lors de la réunion « UBZ » (Ubuntu Below Zero) ayant eu lieu à

Montréal du 30 octobre au 10 novembre 2005, suite à la sortie de la distribution Ubuntu Breezy, réunion préparant le développement de la distribution Ubuntu Dapper.

5.2.4 Canonical

Canonical Ltd. est la société créée par Mark Shuttleworth autour d'Ubuntu. Son objectif « *is committed to the development, distribution and promotion of open source software products, and to providing tools and support to the open source community* »⁵². L'entreprise est une SSLL (Société de Services en Logiciel Libre) offrant principalement des services autour de la distribution Ubuntu, comme le support, la formation à des compagnies offrant du support, l'ajout de fonctionnalités, ... La compagnie embauche un certain nombre de développeurs, travaillant autour de la distribution Ubuntu, ou sur des projets sur lesquels elle souhaite mettre l'emphasis, pour des raisons stratégiques (par exemple Bazaar-NG, voir plus bas section 5.2.8). Elle dispose alors d'un très haut niveau d'expertise dans les différents domaines dans lesquels travaillent ces développeurs. Cette expertise sert à l'entreprise à développer des fonctionnalités spécifiques suivant ses besoins et ceux de ses clients dans les projets des développeurs (Linux, GNOME, KDE, Bzr-NG, etc.), où à répondre à des besoins de correction de bugs suite par exemple à la demande d'un client.

5.2.5 Les liens avec Debian

Debian est une distribution majeure, dont les débuts remontent à 1993⁵³, proposant un « contrat social » avec la communauté du logiciel libre⁵⁴, connue pour sa grande communauté, son attachement au libre, et sa grande stabilité. Toutefois, pour certains,

⁵² Source : <http://www.canonical.com/>

⁵³ Source : http://fr.wikipedia.org/wiki/Debian#Historique_des_versions

⁵⁴ Disponible ici : http://www.us.debian.org/social_contract

elle souffre d'un élitisme important, d'une utilisation trop compliquée pour le néophyte, d'un temps de réaction trop long, ou encore d'une orientation serveur plutôt que desktop.

La distribution Debian, comme les distributions modernes fonctionnent à partir d'un système de gestion de paquets (voir 5.2.6). Ubuntu est une distribution basée sur Debian : à chaque développement de nouvelle version d'Ubuntu – tous les six mois – les paquets en code source de Debian sont récupérés, modifiés et patchés en fonction des différences entre les versions Debian et Ubuntu (étape appelée « *Merge* »), puis une fois cette étape réalisée, les développeurs Ubuntu travaillent sur leur version des paquets, tout en proposant aux développeurs Debian leurs modifications (les patches sont disponibles à travers un système automatisé disponible sur Internet⁵⁵).

Le travail fait par les développeurs d'Ubuntu s'il est intégré par Debian ou par les programmeurs encore en amont, évite au prochain « *Merge* » de devoir refaire les correctifs. Ainsi, le travail est vraiment collaboratif, même avec des entités qui pourraient être vues comme des concurrents.

[...] nous sommes un projet qui tente vraiment de collaborer avec plein d'autres projets - tels que X.org, GNOME, et bien évidemment Debian. Dans beaucoup de cas, le code que nous fournissons est modifié ou différent du code fournit par ces autres projets. Lorsque ça arrive, nous travaillons dur pour s'assurer que nos changements soient publiés le plus largement possible, dans un format qui soit facile à comprendre pour les mainteneurs d'autres projets, et facile à incorporer dans leur propre arbre de travail.

Dans la pratique, nous avons mis fort longtemps pour développer les outils qui rendent facile la collaboration avec Ubuntu, et nous aident à collaborer avec les projets en amont et les autres distributions. Par exemple, nous avons un publicateur automatique de correctifs qui montre aux mainteneurs de Debian quels correctifs pour leurs paquets sont disponibles pour Ubuntu. Il ne pourrait pas être plus facile pour les développeurs de Debian de décider quels correctifs ils veulent, et ceux qu'ils ne veulent pas. Et franchement, c'est bien plus facile

⁵⁵ Disponible sur le site <https://patches.ubuntu.com/>

pour nous si il les prennent effectivement, mais nous ne pouvons pas les forcer. Beaucoup des correctifs n'ont de sens que pour Ubuntu. Comme bénéfice secondaire, ces correctifs sont également disponibles pour Gentoo, Red Hat, Linspire (oui, vraiment) et Suse. Et nous savons qu'ils y jettent un oeil et en utilisent certains, ce qui est cool⁵⁶.

5.2.6 Le système de paquets

Le système de paquetage est un élément central d'une distribution moderne. Ubuntu tire son système de paquetage de la distribution Debian, d'où l'extension des fichiers « .deb ». Ce système permet d'encapsuler dans un paquet le logiciel, la configuration nécessaire au logiciel et les dépendances vis-à-vis d'autres paquets. Les paquets sont regroupés dans des dépôts de paquets, qui peuvent être des supports tels que CD-ROM, intranet, ou encore disponibles sur internet. Grâce à ce système, l'administration du système s'en trouve fort simplifiée. Ce système gestion de paquets de Debian présente l'avantage d'être à la fois simple pour l'utilisateur tout en étant souple et puissant (à peu près n'importe qui ayant des connaissances basiques et suivant un tutoriel est capable de créer un paquet et/ou un dépôt de paquets). L'ensemble logiciel permettant de gérer les paquets, et donc l'installation / désinstallation / dépendances / maintenance du système se nomme APT (pour *Advanced Packaging Tool*)⁵⁷.

⁵⁶ Source : FAQ de Mark Shuttleworth traduite - <http://www.framasoft.net/article624.html>

⁵⁷ « *Advanced Packaging Tool* est un système complet et avancé de gestion de paquets, permettant une recherche facile et efficace, une installation simple et une désinstallation propre de logiciels et utilitaires. Il permet aussi de facilement tenir à jour votre distribution Ubuntu avec les paquets en versions les plus récentes et de passer à une nouvelle version de Ubuntu, lorsque celle-ci sort. APT est un ensemble d'utilitaires utilisables en ligne de commande. Il dispose aussi de nombreuses interfaces graphiques, dont Synaptic, Kynaptic et Adept, et d'interfaces en ligne de commande, comme dselect et Aptitude, afin d'en rendre l'utilisation plus sympathique. ».

Source : <http://doc.ubuntu-fr.org/applications/apt>

Les personnes chargées de créer et de maintenir un paquet sont nommées des mainteneurs (*packager* en anglais). Un ensemble de paquets est présenté sur un support nommé dépôt de paquet.

Plusieurs dépôts de paquets existent. Il est possible pour n'importe qui de créer un dépôt de paquet, toutefois celui-ci sera alors non officiel et nécessitera d'être ajouté manuellement de la part de l'utilisateur.

Les dépôts de paquets officiels sont soumis à des normes de qualité plus strictes – comme gérer pleinement les dépendances de paquets – pour être acceptés par les mainteneurs des dépôts. Pour une version de la distribution les dépôts sont séparés en cinq types :

- la version standard du dépôt, correspondant au moment de la sortie de la distribution,
- « *security* » qui sont des mises à jour de sécurité,
- « *updates* » qui sont des mises à jour de logiciels, mais sans changements majeurs (il s'agit plutôt de correction de bugs),
- « *backports* » qui sont des paquets de logiciels qui ont été créés pour la version suivante de la distribution, mais qui ont été portés sur la distribution précédente afin d'en faire profiter les utilisateurs.
- Le dernier dépôt, « *proposed* » correspond à des paquets proposés, mais considérés comme étant encore en test avant d'être déplacés dans un des autres dépôts.

Par ailleurs, chaque type de dépôt officiel est divisé en quatre branches :

- le dépôt « *main* » composé de paquets libres et supportés par les développeurs d'Ubuntu (le code source doit être disponible, les

modifications et la distribution de copies modifiées doivent être autorisées⁵⁸),

- le dépôt « *restricted* » dont les paquets sont placés dans une zone « contingentée » très réduite, car ils sont supportés, mais non libre (il s'agit principalement de drivers pour utiliser du matériel, nécessaire pour la distribution, mais en tant que paquet non libre). Ils sont sujets à polémique dans la communauté, aussi leur diffusion est-elle régie par prérequis assez strict : pour être intégré dans le dépôt, la licence du logiciel doit autoriser les redistributions, ne pas demander de royalties pour des paiements ou des modifications, doit autoriser le passage des mêmes droits à d'autres logiciels, ne doit discriminer aucune personne, aucun groupe ou champ, ne doit pas être distribué sous cette licence spécifiquement pour Ubuntu, ne doit pas avoir de clause contaminatrice (les autres logiciels fonctionnant avec le logiciel doivent adopter la même licence), doivent permettre à des patches d'être distribués⁵⁹.
- le dépôt « *universe* » est un ensemble de logiciels dont la licence est libre, mais qui ne sont pas supportés par l'équipe d'Ubuntu mais par l'équipe MOTU (pour Master Of The Universe),
- et finalement le dépôt « *multiverse* » qui est un ensemble de logiciels non libres et non supportés par les développeurs Ubuntu (maintenus aussi par l'équipe MOTU)⁶⁰.

Enfin, un dépôt officiel de logiciels commerciaux est maintenu par Canonical.

⁵⁸ Source : <http://www.ubuntu.com/ubuntu/licensing>

⁵⁹ Idem

⁶⁰ Source : <http://doc.ubuntu-fr.org/applications/apt/depots>

Ainsi, Ubuntu offre un compromis en offrant un certain nombre de paquets avec maintien assuré (les paquets supportés), tout en offrant un très vaste choix de logiciels suivi par l'équipe MOTU, gravitant autour d'Ubuntu.

types et branches des dépôts officiels	main	restricted	universe	multiverse
standard				
security				
updates				
backports				
proposed				
Commercial				

Figure 21 - Récapitulatif des différents types et branches de dépôts de paquets

5.2.7 Le fonctionnement par méta-paquets

L'utilisation du système de paquetage de Debian permet un usage intéressant, largement utilisé par la distribution Ubuntu, et faisant d'elle une distribution modulable et réutilisable (voir section 5.2.9, les distributions dérivées). Il s'agit des méta-paquets, qui sont des paquets qui n'installent directement aucun programme, mais possèdent des liens de dépendances permettant d'installer et de garder à jour un ensemble de programmes. Ainsi, Ubuntu s'installe à partir des méta-paquets de base ubuntu-minimal et ubuntu-standard, puis peuvent s'installer par-dessus – en fonction de l'orientation désirée pour la suite de la distribution – les paquets ubuntu-desktop pour installer un bureau GNOME (version Ubuntu traditionnelle), kubuntu-desktop pour installer un bureau KDE (version Kubuntu), ou encore ubuntu-standard, xubuntu-desktop, etc. Cette organisation permet à partir de dépôts divers de construire des distributions modifiées adaptées à des besoins précis, pour créer des distributions d'entreprise ou d'association par exemple, des versions démarrant sur CD-ROM ou clés USB, ...

5.2.8 Quelques projets importants

À partir des orientations stratégiques qu'Ubuntu souhaite suivre, et suivant son mode de développement précédemment exprimé, Ubuntu a mis l'emphasis sur certains projets lui permettant de mieux atteindre ses objectifs. Citons deux projets d'envergures : Launchpad et Bazaar-NG.

Launchpad ⁶¹ est un site internet servant de moyen de coordination pour le développement de la distribution Ubuntu – mais aussi pour d'autres projets / distributions. Le site pourrait en quelque sorte se comparer à un ERP⁶² consacré à de la gestion de projet. Il intègre :

- Malone, un traqueur de bugs;
- un support technique;
- Bounties, un appel à développer des bouts de programmes (éventuellement rémunéré);
- Blueprint, un traqueur de fonctionnalités, permettant d'évaluer et de déposer des spécifications pour les versions en cours de développement;
- Rosetta, une interface web de traduction de logiciels libres (à fin de localisation de la distribution);
- Un gestionnaire de branche Bazaar-NG;
- Un calendrier.

À travers Launchpad, est perçu un effort de rendre le développement de la distribution plus accessible à la communauté et au non-expert (avec par exemple Rosetta, le système permettant de traduire en ligne les applications et ne nécessitant aucune connaissance technique).

⁶¹ <http://www.launchpad.net>

⁶² ERP pour « Enterprise Resource Planning » ou SGI en français (Système de Gestion Intégré)

L'autre projet décrit ici, Bazaar-NG, est un des outils permettant aux développeurs de travailler ensemble sur le même projet. Il s'agit d'un gestionnaire de version, c'est-à-dire d'un logiciel permettant de synchroniser des fichiers sources afin de pouvoir les récupérer, les modifier, en évitant au maximum les problèmes de conflits, quand par exemple deux personnes travaillent en même temps sur le même fichier.

Bazaar-NG – aussi appelé BZR – est un système décentralisé de contrôle des révisions, permettant de créer facilement des branches de projets et de les fusionner ensemble.⁶³ Il s'agit d'« *un système [...] conçu pour être simple à développer et à utiliser. Les systèmes décentralisés de contrôle des révisions donnent aux utilisateurs la possibilité de travailler de manière communautaire via internet. Les utilisateurs du Bazaar-NG décentralisé ont le droit de commiter sur leurs projets libres favorisés sans demander de permission spéciale de la part du projet qui utilisent Bazaar-NG.* »⁶⁴ Il s'agit de faire en sorte que « *ça garde une trace des différentes révisions du code source d'un programme ou des informations similaires et que ça aide les gens à travailler dessus en équipe* »⁶⁵.

Avec Bazaar-NG, Ubuntu permet donc de faciliter l'accès aux développeurs qui voudraient rajouter des fonctionnalités à des logiciels présents dans Ubuntu (en aval), et facilite aussi l'échange d'information avec les versions des logiciels présents en amont (chez les différents développeurs de logiciel, Debian, ...).

Ainsi, grâce au développement de Launchpad et de Bazaar-NG, le développement d'Ubuntu a été simplifié et rendu plus efficace, permettant d'éviter de passer un temps trop important sur des tâches pouvant être automatisées, et permettant une participation plus importante de la communauté.

⁶³ Si le lecteur veut en savoir plus, nous lui recommandons la lecture du fil de conversation <http://linuxfr.org/2005/09/12/19555.html> et les liens proposés.

⁶⁴ <http://bazaar-vcs.org/>

⁶⁵ <http://bazaar-vcs.org/fr/Bzr>

5.2.9 Les distributions dérivées

À partir de la version première d'Ubuntu, plusieurs versions de distributions dérivées ont pu voir le jour, favorisées par le système de méta-paquets (expliqué en section 5.2.7). Parmi les différentes distributions, citons à titre d'exemple :

- Kubuntu, la version d'Ubuntu utilisant l'environnement KDE en remplacement de GNOME;
- Xubuntu, étant une version allégée d'Ubuntu pour les ordinateurs plus anciens
- Edubuntu est la version d'Ubuntu destinée au secteur de l'éducation (écoles, associations, ...)
- Etc.

5.2.10 L'écosystème d'Ubuntu

Suite à la description donnée jusqu'à présent, nous pouvons esquisser sur la Figure 22 l'écosystème d'Ubuntu décrit jusqu'à présent.

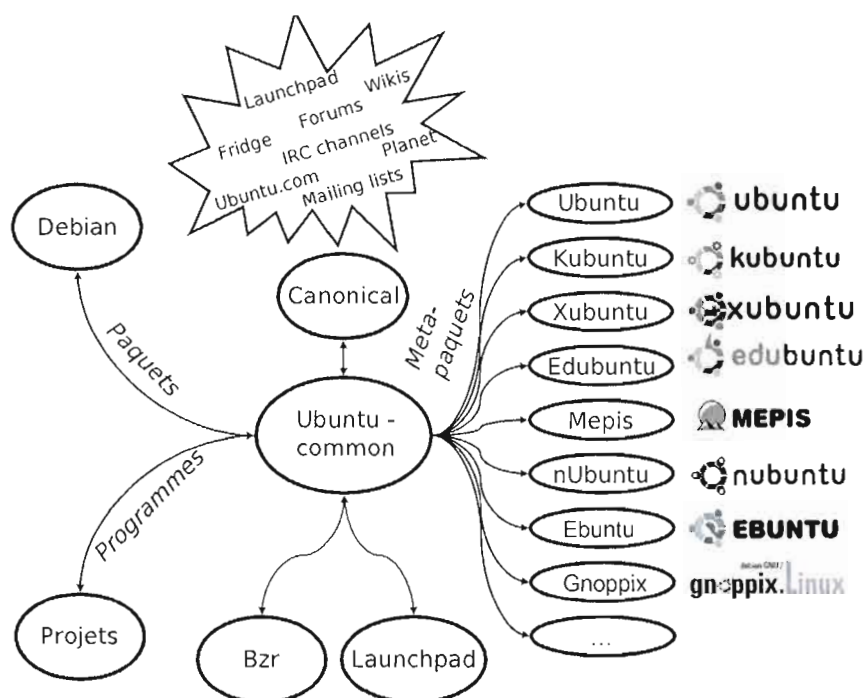


Figure 22 - Écosystème d'Ubuntu

Dans ce chapitre, nous avons présenté les principaux acteurs et entités présents et permettant de comprendre le fonctionnement du projet Ubuntu, à savoir les principaux projets logiciels, les mécanismes de fonctionnement de la distribution, ses valeurs, les bases de son fonctionnement technique, pour aboutir à la représentation schématisée de l'écosystème d'Ubuntu sur la Figure 22. Maintenant que le lecteur est familiarisé à ce terrain, dans le chapitre à venir, nous allons présenter les résultats de notre recherche.

CHAPITRE VI

RÉSULTATS

Dans ce chapitre, nous traiterons de l'étude menée dans la communauté Ubuntu. Nous présentons ici les catégories finalisées, qui ont émergé au fur et à mesure de notre recherche, ont été travaillées, hiérarchisées à partir de notre travail de terrain et influencée par notre travail théorique – tel que cela a été défini dans le chapitre 4 portant sur la méthodologie – jusqu'à l'aboutissement de ce travail de catégorisation présentée ici.

6.1 L'engagement à la communauté

Les membres de la communauté, malgré une grande diversité (section 6.1.1) partagent des valeurs communes (section 6.1.2) qui les maintiennent ensemble au sein de la communauté.

6.1.1 Une grande diversité des contributeurs

La communauté est composée de nombreux individus aux origines géographiques différentes et provenant de milieux divers. Pour illustrer cette diversité, nous citerons à titre d'exemple <A>, membre de la communauté GNOME, venant de terminer un mémoire sur les grilles de calculs⁶⁶, , développeur de logiciel destiné à Ubuntu est membre d'une association promouvant une « *culture alternative et militante. Nos combats ? La liberté, l'autogestion, la culture pour tous, les luttes anti-autoritaires et*

⁶⁶ Source: Blog agrégé par Planet Gnome-fr

anti-fascistes, l'anti-capitalisme »⁶⁷, <C> étudiant en master de biologie informatique⁶⁸, <D>, doctorant dans la conception et la modélisation informatique⁶⁹, <E>, « *spécialiste de la sécurité dans Internet et de la cryptographie* »⁷⁰, <F>, contributeur à plusieurs projets, chercheur en physique⁷¹, <G> travaillant dans la production mécanique, <H>, coordonnateur dans une entreprise multinationale⁷², <I>, consultant IT⁷³, <J>, chercheur au MIT en publication électronique⁷⁴, <K> étudiant en droit⁷⁵, <L> chercheur en mathématique⁷⁶, <M> étudiant en art⁷⁷, <N>, musicien professionnel⁷⁸, etc.

Cette diversité de profils et d'approche est la source d'une grande diversité de connaissances dans la communauté, de personnes réunies autour d'un même objectif de contribution à l'avancement du logiciel libre.

Comment une telle diversité de contributeurs, de provenances et de trajectoires si différentes est-elle amenée à travailler ensemble? Il y a certes la passion de la technique menant souvent à l'expertise (section 6.2.2), et le rythme effréné de l'innovation qui en fait un élément passionnant, d'autant plus que chacun peut y contribuer selon ses choix (section 6.2.3), mais il y a aussi un ensemble de valeurs fondamentales partagées.

⁶⁷ Source: Blog agrégé par Planet Ubuntu-fr

⁶⁸ Source: Blog agrégé par Planet Ubuntu-fr

⁶⁹ Source: Blog

⁷⁰ Source: Blog agrégé par Planet Ubuntu-fr

⁷¹ Source: Blog agrégé par Planet Ubuntu-fr

⁷² Source: Blog agrégé par Planet Ubuntu-fr

⁷³ Source: Blog agrégé par Planet Ubuntu

⁷⁴ Source: Blog agrégé par Planet Ubuntu

⁷⁵ Source: Blog agrégé par Planet Ubuntu

⁷⁶ Source: Wiki Ubuntu

⁷⁷ Source: Wiki Ubuntu

⁷⁸ Source: Wiki Ubuntu

6.1.2 Des valeurs partagées

Pour fédérer cette communauté, nous constatons un ensemble de valeurs partagées par celle-ci. Les contributeurs se sentent généralement profondément engagés, comme l'illustre le projet Tribu dans le message à l'origine du projet :

Cela dit derrière linux et particulièrement, Ubuntu, il y a une éthique, un esprit de partage et à cela j'y suis attaché. Contrairement à ce que je peux laisser croire, j'aime Ubuntu et je suis fier de ses progrès. Mais demain je veux le voir vivre et cela passer-a par un service `.ubuntu` comme `.mac`⁷⁹

La communauté partage un ennemi commun, comme le laisse transparaître le Bug #1 (cf. section 5.1.2) à l'initiative de Mark Shuttleworth, proposant un système d'exploitation simple et humain, capable de faire concurrence au système d'exploitation Microsoft™ Windows. Nous retrouvons par exemple chez un développeur important d'Ubuntu accolée à sa signature la phrase « *In a world without walls and fences, who needs Windows and Gates?* », jouant sur le double sens des mots, *Windows* faisant référence au système d'exploitation de Microsoft™ et à la signification anglaise (fenêtre), et *Gates* au créateur du système d'exploitation précédemment cité et à la traduction « porte ». Le système d'exploitation concurrent est ici clairement assimilé à un système fermé et limitant, imbriquant l'utilisateur, Ubuntu (et les autres systèmes libres) permettant d'éliminer symboliquement les murs et obstacles existants. Notons que Microsoft™ représente traditionnellement dans la communauté du libre l'ennemi sur lequel se cristallise la pulsion de mort des utilisateurs⁸⁰.

Une des valeurs centrales d'Ubuntu est l'humanisme affiché, comme son propre nom le laisse suggérer, ou encore la thématique artistique de la distribution qui est principalement de couleur orange et marron, des couleurs chaudes rappelant la

⁷⁹ Fil de discussion de Tribu sur le forum Ubuntu-fr

⁸⁰ Cf. Ferrary, Pesqueux, p.262

volonté d'humanité, d'humaniser l'informatique. Ces choix ne sont pas anodins, comme le dit Shuttleworth dans sa FAQ :

Pourquoi le fond d'écran par défaut d'Ubuntu est-il MARRON ?

Le thème général de la première série de version d'Ubuntu est "Humanité". Ca détermine notre choix graphique autant que notre sélection de paquets et les décisions autour de l'installateur. Notre thème par défaut dans les quatre premières versions d'Ubuntu est appelé "Human", et il met l'accent sur la couleur humaine chaude - le marron.

Oui, c'est plutôt inhabituel dans un monde où la plupart des fonds d'écran sont bleus ou verts, et MacOSX est devenu comme des ustensiles de cuisine. En partie, nous aimons le fait qu'Ubuntu soit différent, plus chaud. L'ordinateur n'est plus un outil, c'est une extension de votre esprit, votre porte vers les autres (par email, voip, irc et le web). Nous voulions une sensation qui soit unique, saisissante, apaisante et surtout, humaine. Nous avons choisi le marron. C'est plutôt un choix hautement risqué, car pour restituer le marron votre écran doit restituer de subtiles nuances de bleu, et de vert, et de rouge. Même de légères variations par rapport à la norme peuvent modifier le "marron" substantiellement. Mais les moniteurs et les écrans LCD sont de nos jours de plus en plus à un niveau qui nous a fait ressentir le risque comme acceptable. Dans Hoary et Breezy nous apportons un brun plus riche, plus rouge, basé sur les commentaires des utilisateurs d'ordinateurs portables bas de gamme et d'écrans LCD⁸¹.

Par ailleurs, le fait que bon nombre de développeurs ainsi que le créateur, Mark Shuttleworth tiennent des blogs et soient relativement accessibles – dans les commentaires, *mailing-lists* ou par courriel – contribue à ce sentiment de proximité avec les utilisateurs. Cela contribue à souder la communauté, qui à travers le code de conduite (be considerate, be respectful, be collaborative) pousse vers un certain nombre de valeurs, dont les principaux développeurs sont les représentants les plus symboliques. Shuttleworth résume ainsi ces valeurs d'Ubuntu :

Essentially, we want to create a sustainable, practical home for the very best Linux engineers and architects (initially drawing

⁸¹ Source: traduction sur <http://www.framasoft.net/article624.html#10> de <https://wiki.ubuntu.com/MarkShuttleworth>

primarily from the pool of Debian developers but now we're starting to bring in upstream folks too). [...] Ubuntu will have a role to play in making Linux widely and freely available and keeping it at the cutting edge. Our job is to make the amazing, cutting edge work of thousands of free software developers available in a neat, elegant package that anybody can deploy free of charge, with easy access to the whole universe of free software, and for which they can get commercial grade support if they want it⁸².

Plus généralement, les valeurs « *hackers* » sont à la base d'Ubuntu. Raymond résume la culture *hacker* en précisant que « *les hackers résolvent des problèmes, ils construisent, et ils croient en la liberté et en l'assistance mutuelle bénévole* »⁸³. Est aussi associée à la culture *hacker* la volonté de bidouiller, comme pour faire fonctionner des choses ensembles qui n'étaient pas prévues fonctionner ensemble à la base, comme l'illustre le message suivant :

Apparently, I'm looking for a device designed for a very peculiar niche. If so, I figure the lazyweb is the most likely place to find answers about it: I'm looking for a scanner with built-in fax functionality, but without the laser or inkjet printer element of the usual "multifunction" hardware suite. Yeah, yeah - faxes suck, but your own fax sucks less than someone else's hunk of junk.

I have a perfectly functional printer, so the design would be something like a flatbed scanner with an inbuilt modem and fax keypad - stick some paper on the scanner, and dial away. The ideal design would have to include ethernet for printer connection and network faxing or scanning.

Sure, I could rig up a flatbed scanner and a few bits of software, but I want to see whether or not this sucker exists first. Looking for a solution, not a science project. ;) Of course, if it doesn't exist, I'll start banging some of my favourite science project rocks together... SIP phone + Asterisk + network-capable flatbed scanner + hacking == rock!

Update: SIP phone + Asterisk doesn't necessarily mean VoIP (which, as a commenter points out, is a shitty way to fax)... I'd set up a prefix to kick off the scan-and-fax process, sending

⁸² Mark Shuttleworth, 3 août 2006, <http://www.markshuttleworth.com/archives/49>

⁸³ « Comment devenir un hacker », Eric S. Raymond, <http://www.freescap.eu.org/eclat/3partie/Raymond2/raymond2txt.html>

```
the scanner input to the number dialled on the SIP phone. Just
like a fax machine, only with distributed hardware components.
;)84
```

La liberté est aussi une valeur importante dans Ubuntu, comme le montre le cas présenté dans la section 6.2.4 où l'intégration d'un logiciel offrant une fonctionnalité importante n'a pas été possible tant que la licence d'une des librairies utilisées ne fut pas changée en une licence libre.

Le partage est aussi l'une des valeurs promulguées dans la communauté, comme en témoigne ce message accompagnant un programme visant à améliorer la distribution Kubuntu :

```
[TheKubuntu :] Petit script qui permet de finaliser
l'installation de Kubuntu C'est un petit script que je me suis
créé pour mon utilisation personnelle, alors pourquoi pas le
partager!?85
```

Ou encore cet autre utilisateur, à propos d'une fiche technique qu'il a écrite :

```
Ce billet n'a pas vocation d'être une documentation de référence
ou un tutorial. C'est juste un petit aide mémoire à l'origine
juste pour moi, mais qui peut sans doute être utile à
d'autres86.
```

Le partage d'information est très important, comme l'illustre cette discussion sur la *mailing-list* des traducteurs francophones, contribuant à travers ce partage d'information à l'émergence d'une nouvelle façon de faire, d'une nouvelle connaissance :

```
<A> Je viens de voir l'erreur suivante dans KMail, lors d'une
<A> tentative de décryptage d'un mail par GnuPG :
<A> [...]
<A> Erreur : Mauvait mot de passe
<A>
<A> MauvaiT => Mauvais
```

⁸⁴ Source: Blog agrégé par Planet Ubuntu

⁸⁵ Source: Blog agrégé par Planet Ubuntu-fr

⁸⁶ Source: Blog agrégé par Planet Ubuntu-fr

```

<A>
<A> J'ai cherché en vain dans Rosetta... Est-ce que KMail a été
<A> importé, ou bien faut-il remonter le pb à l'équipe de
<A> traduction de KDE ?

<B> [...] L'erreur est dans libpgp-error.po
<B> En plus il ne doit pas y avoir de majuscule après un « : »
<B> (valable aussi pour la ligne du dessus) [...]

<A> [...] Pour que je m'améliore : comment as-tu fait pour
<A> retrouver ça ??? [...]

<B> C'est une astuce dont <D> m'avait fait part:
<B> Tu télécharges les sources .po sur ton pc en faisant :
<B>
<B> apt-get source language-pack-kde-fr-base
<B> [...]
<B> etc. Aide-toi de la complétion automatique pour avoir les
<B> noms des paquets source.
<B> [...]
<B> Tu t'y mets en ligne de commande et dans un terminal tu
<B> saisis :
<B>
<B> grep -n -R "Mauvais mot de passe" *
<B>
<B> Attention c'est sensible à la casse.
<B>
<B> Après le plus simple c'est de télécharger le .po depuis
<B> rosetta pour trouver le numéro de la chaîne dans Rosetta
<B> (avec kbabel par exemple). [...]

<C> Tu peux ajouter l'option -i à grep pour ne pas tenir compte
<C> de la casse87.

```

La communauté est aussi imprégnée d'autodérision. Citons à titre illustratif, le *hackergotchi* qui est « *a picture of a writer used as an icon to identify the author of a given RSS feed in blog aggregators* »⁸⁸, représentant une forme d'autodérision de l'image des contributeurs. Des *hackergotchis* de la communauté GNOME sont représentés en Figure 23. La communauté Ubuntu sur son *planet* dispose aussi de ses *hackergotchis*. Donnons un autre exemple de dérision de la communauté, aux rencontres mondiales du logiciel libre, où le dernier intervenant donne une conférence

⁸⁷ Source: *mailing-list* de traduction francophone

⁸⁸ Source: <http://en.wikipedia.org/wiki/Hackergotchi>.

« *Le manchot descend-il du singe? L'exemple de Linux* » où la thèse défendue avec humour par le présentateur est que les programmeurs Linux descendent du manchot⁸⁹, tournant en dérision la communauté du libre et le monde de la recherche⁹⁰.

La culture *hacker* est élitiste, comme le laisse entendre Raymond lorsqu'il décrit les *hackers* comme pouvant se considérer *hacker* si d'autres personnes ayant contribué à cette culture le reconnaissent comme tel (cf. section 2.2). Nous n'avons pas ce phénomène chez Ubuntu, au contraire, la communauté tente de diffuser les valeurs *hackers* avec la volonté de mise à disposition du plus grand nombre, une forme de démocratisation de ces valeurs se traduisant par exemple avec la mise en place du NUN pour *New Users Network*, dont le but est :

to try and help new Ubuntu Users get to grips with Ubuntu. Members of the New User Network will spend a lot of time on IRC, the forums and the mailinglists⁹¹.



Figure 23 - Exemples d'*hackergotchis* de la communauté GNOME⁹²

⁸⁹ Le manchot est le symbole de linux

⁹⁰ Source: <http://free-electrons.com/pub/video/rml2006-sebastien-blondeel-pingouin-singe.ogg>

⁹¹ <https://wiki.ubuntu.com/NewUserNetwork>

⁹² Source: <http://planet.gnome.org/heads/>

6.2 Un fonctionnement mérito-démocratique

L'une des principales nécessités dans une démocratie est la possibilité donnée à tous de s'exprimer librement, pour être capable de faire un choix éclairé (6.2.1). Toutefois, certaines voix ont bien entendu plus de poids que d'autres, et ceci se fait principalement selon la reconnaissance de la personne, elle-même découlant de son expertise et de son niveau de contribution (6.2.2). À partir de ces prises de position pondérées se font alors des choix sur le mode de fonctionnement lui-même (6.2.3), entraînant la création de la connaissance à travers l'amélioration de la distribution (section 6.3).

6.2.1 Le droit à la parole

Dans la communauté, tout le monde a le droit de s'exprimer ouvertement, cela est même encouragé. Ainsi, suite à la quantité de travail nécessaire dans Dapper, Mark Shuttleworth propose un grand débat sur l'acceptation d'un retard de six semaines pour affiner le travail :

Hi all

I'm writing to propose a six week delay in the release date of Dapper, in order to do additional validation, certification, localisation, and polish. I would like to call for a community "town hall" meeting on Tuesday 14th March - once at 09:00 UTC (for the Aussies and Asian communities) and then again at 18:00 UTC (for Europe and the Americas). The meetings will be in #ubuntu-meeting on irc.freenode.net. Based on feedback at those meetings, we will ask the Tech Board and the Community Council to take a view on the proposal, and announce the decision by the end of the week.⁹³

Comme nous l'avons vu section 5.2.3, il y a différents lieux de discussion : coexistent l'IRC offrant la possibilité d'être à plusieurs en temps réel, les *mailing-lists* permettant des échanges plus argumentés, le temps y étant alors discret. Ces dernières

⁹³ Source: *mailing-list* ubuntu-dev-announce

de par la possibilité de s'abonner et de mieux structurer l'information représentent un lieu avec un auditoire plus important, où les décisions importantes sont abordées. Les blogs et forums peuvent aussi servir aux débats d'idées. Les forums quant à eux, sont plus utilisés par les utilisateurs que par les développeurs. Enfin, les blogs permettent un point de vue plus personnel, et une meilleure connaissance de la personne.

Des discussions sur des fonctionnalités ont souvent lieu sur la *mailing-list* des développeurs, partant d'une idée, d'une remarque qu'un contributeur s'est permis de faire et que d'autres n'hésitent pas à commenter :

```
<A> I've noticed as I work that it would be absolutely great
<A> to have the workspace switcher have an option of setting the
<A> Desktop. Something like this:
```

```
<A>
```

```
<A> I have a home Desktop but I also have a project I'm working
<A> on which I'll put on workspace 3. If I'm on workspace 1,2
<A> or 4 I see my home Desktop, however, if I've switched to
<A> workspace 3, I see only the files on the Desktop that are
<A> associated with my project. Only the files I have put there.
```

```
<A>
```

```
<A> This seems like a pretty simple thing and I know some of
<A> this has been discussed before, specifically custom
<A> backgrounds for each workspace. This of course would go
<A> beyond that.
```

```
<A>
```

```
<A> Comments
```

```
<B> It's an interesting idea, but I think it could only ever
<B> be an optional, non-default feature, as a typical user
<B> would become confused at their files disappearing if they
<B> switch desktops (especially inadvertently).
```

```
<A> Yes, I think it would be an option in say the workspace
<A> switcher which would talk to Nautilus via dbus or something
<A> similar. It would be very cool though. I find that
<A> keeping a bunch of files on the desktop that I am currently
<A> working on, even if they are sym links is extremely
<A> seful.94
```

⁹⁴ Source: *mailing-list* ubuntu-devel

Les fonctionnalités donnent souvent lieu à des échanges forts construits, argumentés et bien illustrés entre plusieurs contributeurs, qui savent que les fonctionnalités effectivement ajoutées émergeront des nouvelles idées et façons de faire de la discussion et du fil d'échanges :

```

<A> Hi!
<A> I noticed that Trash/Recycle Bin/Wastebasket isn't completed
<A> "application" on any platform. Biggest problem with trash is
<A> that it gets filled and rarely gets purged.
<A>
<A> Shouldn't Trash purge automatically after some period of time?
<A> Let's say that, by default, it never get's clean, but user
<A> should be able to define some amount of time when Trash
<A> get's automatically cleaned. On every startup applet should
<A> check for files older then N days and remove them if user
<A> wants for files to stay in Trash for N days. On the other
<A> hand, some of use don't log out for couple of days :)
<A> Maybe some other way of implementation?
<A> [...]

<B> Actually my beleif is that it should be emptied when the
<B> system is low on harddrive space. That way the user will
<B> never actually have to bother about empying the trash.
<B> Actually I also the /var/cahe/apt/archives should also be
<B> purged when harddrive space is getting rare. Maybe there
<B> should be a centralized mechanism that gets called back
<B> when the user attains a given level on her harddrive.

<C> I don't agree with the disk-space dependent behavior. That
<C> would defeat the purpose of "Trash", from the trash feature
<C> you expect to be able to recover from a delete operation at
<C> least for a small amount time, it should be seen as a data
<C> safety mechanism and not as "nice to have" based on the disk
<C> free space.
<C>
<C> In my opinion, the lack of disk space should require some
<C> investigation by the user, deleting data from trash/cache
<C> could be a tip, but not something to be done automated.95

```

Ainsi dans l'exemple sus-mentionné, au court de la discussion, de nouvelles possibilités émergent des différences de points de vue, comme, où la possibilité

⁹⁵ Source: *mailing-list ubuntu-devel*

Les caractéristiques généralement associées aux trolls sont d'être de « *querelles futiles* »⁹⁶, « *de mélanger des discussions aux thèmes opposés* »⁹⁷ de provoquer des « *flame war (guerre de messages incendiaires) entre des interlocuteurs qui ne devaient pas, à l'origine, discuter ensemble (car leurs préoccupations sont différentes)* »⁹⁸, etc.

Les trolls donnent parfois lieu à des débats de valeurs, arrivant à l'insulte des partenaires – c'est pour cette raison qu'ils sont généralement aussi mal vus – mais ils donnent aussi lieu à une forme de discussion argumentée, remontant souvent jusqu'aux différences de valeurs et de visions des participants, remontant alors jusqu'à la source de la divergence d'opinions et donnant alors la possibilité à un compromis, une synthèse parfois nouvelle (et alors créatrice de connaissance) le débat étant souvent fort instructif.

Ainsi, illustrons ce fait avec le projet qui a débuté suite à un message qu'un des protagonistes interviewés, <F>, qualifie de « *post sur le forum qui ressemblait à un beau troll* :) ». Le message écrit par <A> disait en effet :

Il ya quelques jours je posais une question sur ce forum afin de réussir à synchroniser Evolution avec icalx.

Pas une réponse ! Et oui l'ubuntiste franchouillard est plus préoccupé par booster son firefox, optimiser sa mule pour télécharger le dernier nanar made in us. Bref de la bidouille stérile.

[...]

L'avenir de l'informatique familiale il est là, Apple a tout compris. Le combat entre OS c'est déjà du passé. Demain à peu de différences près les OS seront tous sécurisés, performants et userfriendly. Alors le bon père de famille il se foutra total d'avoir Vista ou Ubuntu par contre il ira vers celui qui lui offre des services.

[...]

⁹⁶ Source: Blog agrégé par Planet Ubuntu-fr

⁹⁷ Source: « Qu'est-ce qu'un troll ? » par par Lirresponsable, Pascale Louédec et ARNO*, uZine, <http://www.uzine.net/article1032.html>

⁹⁸ idem

Pour conclure, si la communauté ubuntu ne développe pas ce service, vous pourrez dire adieu à votre distribution qui sera peut être l'os le plus performant mais qui restera bien au chaud dans la chambre d'un informaticien bidouilleur⁹⁹.

<A> toujours, reconnait plus tard :

Pour que ce post au titre un peu provocateur ouvre la voie vers un projet constructif je résume ici la réflexion qui se développe au cours des différentes interventions¹⁰⁰.

Certains trolls reviennent fréquemment, ayant une fonction implicite de représentation des différences de visions relatives des participants remplissant un rôle symbolique au sein de la communauté. Il en est souvent ainsi du troll GNOME/KDE, qui est capable de donner lieu à des discussions endiablées, mais toujours entre participant aux logiciels libres, s'affirmant tous dans cette culture, et finalement sachant ne pas être véritablement opposés :

```
(23:12:34) <E>: j'ai pensé aussi qu'on pourrait faire un plug-
                in pour gnome-vfs qui nous permettrait
                d'accéder aux données à partir de nautilus
(23:12:55) <E>: ça n'a pas l'air très compliqué, il y a un bon
                tuto
(23:13:03) <G>: sur Kde ca devrait pas etre mal non plus
(23:13:12) <G>: vu le niveau d'integration de toutes les applis
(23:13:16) <E>: j'utilise pas Kde :s
(23:13:29) <E>: je sais pas comment ça marche
(23:13:33) <G>: lol
(23:13:37) <E>: mais oui, si quelqu'un est motive
(23:13:48) <G>: fait pas ton noob windosien :)
(23:14:14) <E>: non je suis un extrémiste gnome
(23:14:25) <G>: mouarf
(23:14:30) <E>: tout ce qui a besoin de la libQt je ne
                l'installe pas :)
(23:14:47) <G>: j'utilise gnome aussi mais suis un accro de
                Amarok et k3b :)
(23:14:50) <E>: tout ce qui commence par k non plus (sauf kernel
                :) )
(23:15:00) <G>: MDR
(23:15:29) <E>: c'est historique ... mais bon je n'ai rien
                contre kde
(23:15:37) <E>: bref101.
```

⁹⁹ Fil de discussion de Tribu sur le forum Ubuntu-fr.

¹⁰⁰ Fil de discussion de Tribu sur le forum Ubuntu-fr.

Ainsi, ces trolls offrent aux participants des lieux de socialisation. Linus Torvalds, créateur du noyau Linux donne souvent aux libristes l'occasion de troller, par exemple en déclarant sur la *mailing-list* de GNOME « *I personally just encourage people to switch to KDE* » (voir section 5.1.4 pour la citation complète). À propos d'une modification d'architecture proposée par Intel :

```
This other Intel brain-damage (the first one being ACPI) [...]Of
course, it's somewhat questionable whether people have actually
gotten smarter or stupider in the last 30 years. It's not enough
time for evolution to have increased our brain capacity, but it
certainly is enough time for most people to no longer
understand how hardware works any more102.
```

Torvalds toujours, à propos d'un choix technique fait sur FreeBSD, un noyau libre « concurrent » de Linux :

```
That's a huge mistake, and anybody that does it that way
(FreeBSD) is totally incompetent.
```

```
I claim that Mach people (and apparently FreeBSD) are
incompetent idiots103.
```

Il s'agit d'une volonté de lancer des débats que certains ont d'ailleurs remarquée :

```
I think you're taking this too personal. I think Linus' seeming
flame bait here is intended to make people think & respond. Once
debate get's too academic and dry, lots of people don't care as
much. But state things like this and you'll get feedback :)104.
```

Nous l'avons dit, si les trolls sont mal vus par la communauté, c'est parce qu'il arrive qu'ils dégénèrent, donnant lieu à des débats intensément passionnels, quittant alors l'argumentation logique, et ne s'appuyant plus sur aucun fait autre que les messages précédents. Un exemple dans la communauté Linux est celui de l'intégration du système de fichier Reiser4 au sein du noyau. Il s'agit d'un système de fichier, qui

¹⁰¹ Source: discussion instantanée à propos du projet TribU.

¹⁰² Source: <http://kerneltrap.org/node/6884>

¹⁰³ Source: <http://kerneltrap.org/node/6506>

¹⁰⁴ idem

pour son créateur est extrêmement performant, et intègre des innovations majeures¹⁰⁵. Le système de fichier n'a toutefois pas été intégré en juin 2005, car pour le mainteneur de la branche des systèmes de fichiers,

```
<A> The plugin stuff is crap. This is not a filesystem but a
<A> filesystem + new layer. IMO considered in that light, it
<A> duplicates functionality elsewhere.
```

Chose non acceptée par le créateur de Reiser4 :

```
<B> What functionality where? Please remember that this is per
<B> file, per item, per node, per attribute, per disk format,
<B> per bitmap, per super block, etc., abstracting, not per
<B> filesystem abstracting.
<B> [...]
<B> Please reconsider your view.
```

```
<C> [...] we had this discussion before. And the consensus was
<C> pretty simple: the disk structure plugins are your business
<C> and fine to keep. The higher-level plugging that just add
<C> another useless abstraction below file_operation /
<C> inode_operation / etc. are not. keep the former and kill
<C> the latter and you're one step further. [...]
```

```
<B> [...] Let us tinker with our FS, and you tinker with yours,
<B> and so long as what we do does not affect your FS, let the
<B> users choose.
<B> In the end, somebody will write a new fs that steals the
<B> good ideas from both of us, and obsoletes us both. They can
<B> only do this though if we are left to be both free to
<B> implement differing filesystem designs.
<B> I do not tell you how to design XFS, why are you making my
<B> life unpleasant? [...]
```

```
<A> [...] You're basically implementing another VFS layer inside
<A> of reiser4, which is a big layering violation.
<A> This sort of feature should -not- be done at the low-level
<A> filesystem level. [...]
[...]
```

```
<B> [...] There's been sloppy code in the kernel before. I
<B> remember one bit in particular which was commented "Fuck me
<B> gently with a chainsaw." If I remember correctly, this had
<B> all of the PCI ids and the names and manufacturers of the
<B> corresponding devices -- in a data structure - in C source
<B> code. It was something like one massive array definition,
```

¹⁰⁵ Source: <http://www.namesys.com/benchmarks.html>

 or maybe it was a structure. I don't remember exactly,
 but...
 [...]

<A> Existence of ugly code is not an excuse to add more¹⁰⁶.

 a finalement décidé de modifier son code pour intégrer le système de fichier au noyau. Après plus d'une année de modifications, fin juillet 2006 un nouvel échange a lieu sur la *mailing-list* des développeurs du noyau :

 [...] Plugins --do not-- mean that you can just change the
 filesystem format willy-nilly, with zero impact.
 [...]
 <A> Yes they do.....

<D> [...] Here's a few ground rules at least from my viewpoint:
 <D> - as long you call them "plugins" and treat them as such, I
 <D> (and I suspect a lot of other people) are totally
 <D> uninterested, and in fact, a lot of people will suspect that
 <D> the primary aim is to either subvert the kernel copyright
 <D> rules, or at best to create a mess of incompatible semantics
 <D> with no sane overlying rules for locking etc.
 <D> - the kernel does not, and *_will_* not, support "hooks" that
 <D> aren't actually used (and make sense) by normal kernel uses,
 <D> for largely the same reasons. Interfaces need to be
 <D> architected, make sense, and have real and valid GPL'd uses.
 [...]

Au bout d'un long échange, <E>, un développeur influent de par ses responsabilités au sein du projet (il est mainteneur d'une branche importante du noyau) intervient :

<E> Meanwhile here's poor old me trying to find another four
 <E> hours to finish reviewing the thing.
 <E>
 <E> The writeout code is ugly, although that's largely due to a
 <E> mismatch between what reiser4 wants to do and what the
 <E> VFS/MM expects it to do. If it works, we can live with it,
 <E> although perhaps the VFS could be made smarter.
 <E>
 <E> I'd say that resier4's major problem is the lack of xattrs,
 <E> acls and direct-io. That's likely to significantly limit
 <E> its vendor uptake. (As might the copyright assignment
 <E> thing, but is that a kernel.org concern?)
 <E>

¹⁰⁶ Source: <http://kerneltrap.org/node/5330>


```
<E> The plugins appear to be wildly misnamed - they're just an
<E> internal abstraction layer which permits later feature
<E> additions to be added in a clean and safe manner. Certainly
<E> not worth all this fuss.
<E>
<E> Could I suggest that further technical critiques of reiser4
<E> include a file-and-line reference? That should ease the
<E> load on vger.
```

```
<F> [...] I took a quick look at it and I must say that most of
<F> the things that tripped me up when I first looked at it a
<F> long time ago are gone now107.
```

En étudiant attentivement le code source, <E> a ici été capable de recadrer le débat, l'enracinant dans les faits et le rendant à nouveau raisonnable, alors qu'il dégénérât suite à la trajectoire qu'il avait prise un an auparavant. Finalement, la fonctionnalité sera certainement intégrée dans les prochaines versions du noyau.

Dans la communauté Ubuntu, les trolls dégénérant ainsi en insultes restent rares, de par la charte Ubuntu qui spécifie :

```
Be respectful. The Ubuntu community and its members treat one
another with respect. Everyone can make a valuable contribution
to Ubuntu. We may not always agree, but disagreement is no
excuse for poor behaviour and poor manners. We might all
experience some frustration now and then, but we cannot allow
that frustration to turn into a personal attack108
```

afin d'éviter au maximum les problèmes tels ceux relatés dans le cas précédent. Ainsi, la réponse de au message suivant posté sur la *mailing-list* des développeurs illustre cette volonté :

```
<A> [...] I guess the apple interface thing is hard to argue
<A> with. Every major apple enthusiast I know (i.e. people who
<A> think any non-MacOS UI is very poorly designed) has an IQ
<A> below the room temperature in celcius109.
```

¹⁰⁷ Source: <http://kerneltrap.org/node/6922>, l'emphasis est de nous.

¹⁰⁸ Source: <http://www.ubuntu.com/community/conduct>

¹⁰⁹ Source: *mailing-list* ubuntu-devel

 Please read (and follow!) the code of conduct when posting
 to this list ...¹¹⁰

Des divergences d'opinions ont toutefois bien lieu, sur la *mailing-list* ou encore à travers les blogs des développeurs (agglomérés par planet)

<A> It is a common belief that Ubuntu provides newer software
<A> than Debian. This is of course true when Ubuntu is compared
<A> to Debian stable. This is also true for specific software
<A> packages, like Gnome. But how does it compare in general ?
<A> [...]
<A> So Debian easily wins
<A> [...]¹¹¹

 I'm confused by your blog, <A>
 There seems to be no point to it; you're comparing the
 stable releases of one distribution against the unstable
 archive of another one with a completely different release
 schedule.
 Of course the software is going to be out of date! [...] ¹¹².

Notons que même sans participer, le fait de suivre des débats apprend beaucoup au lecteur à travers les argumentations respectives, qui constitue un bon vecteur de transfert de connaissance de par la possibilité d'accéder à sa phylogénèse :

<A> Debian has been around for ages, as have some of it's
<A> problems: "Hard" to administrate [...] So, I was more than
<A> happy to see installing hoary was a breeze and it simply
<A> worked. Part is to attribute to improvements on the
<A> foundations of linux but also a lot is hard work on ubuntu's
<A> part.
<A> [...]
<A> And why does ubuntu have to cater the servers? Doesn't
<A> debian do this **very** good? Why do you need 2.6.15 on a
<A> server? I think an admin should even be kicked for using
<A> this on a server! Leave debian a little breating space and
<A> the collaboration will be lots better.

 Being a good server platform is not the same as what we
 are doing. Debian is great for server admins because they
 can install just about anything they want and tweak it to
 their hearts content.

¹¹⁰ idem

¹¹¹ Source: Blog agrégé par Planet Ubuntu

¹¹² Source: Blog agrégé par Planet Ubuntu

 Ubuntu-Server is a whole different type of server support.
 Our intention is to get hardware certification. Something
 that has been hard to do for Debian (through no fault of
 their own).

<A> That I don't really understand. We are talking linux here,
 <A> as in kernel, right? I know that ubuntu, as debian, as
 <A> fedora, suse and whoever, all have their patches that they
 <A> apply to the vanilla kernel from kernel.org. I don't like
 <A> this very much since it spreads the whole development/patch
 <A> thing all over the place but Linus likes it and his word is
 <A> law. But why does this certification have to be for the
 <A> ubuntu kernel - instead of the debian kernel?

<A> [...]

<A> I mean, to some win admin or old-school unix admin a
 <A> "certified" distro might look sexy and might legitimate his
 <A> decision to spend 1000\$ on rhe, but we, you and - I hope -
 <A> me, know that this is rubbish.

 Again, you are over simplifying. No matter what you do, each
 distro has it's own subtleties.

<C> Is it me or is there a certain shift from "linux for human
 <C> beings" to "linux for the corporate environment - free!". I
 <C> would think that already enough companies cater for the
 <C> server side of linux: ibm in conjunction with ..., novell
 <C> and such that in that area there are enough players. Like I
 <C> said, and microsoft said, the licensing cost is - for a
 <C> company - neglectable to the cost of support and the cost of
 <C> e.g. oracle.

<D> Well, I would comment this with "Linux is Choice" and there
 <D> is never enough of choice :) And, seriously, SysAdmins are
 <D> as well Human Beings. Sometimes this human nature can be
 <D> hide, but after 10-12 hours of work, the SysAdmin becomes
 <D> again human.

[etc. (À peu près 90 messages)]¹¹³

6.2.2 L'expertise technique

Tous les contributeurs, nous l'avons vu avec <E> dans le cas de Reiser4 (section précédente) ne bénéficient pas du même niveau de reconnaissance. Ce niveau de reconnaissance est fonction de l'importance des contributions de ces individus. La

¹¹³ Source: *mailing-list* ubuntu-devel, l'emphase en gras est de nous.

technique a une place très importante dans le logiciel libre, d'autant plus que son usage est particulièrement facilité par rapport à d'autres domaines, tout le monde ayant accès aux outils et aux données de par la nature même du logiciel à code source ouvert, d'où l'importance qui lui est accordée.

Ces références techniques peuvent aussi servir pour des conseils, et pour l'ajout de fonctionnalités à l'initiative de l'utilisateur pour ses propres besoins, leur permettant ainsi d'apprendre et d'améliorer leurs connaissances et leurs pratiques du système :

```
<A> OpenOffice.org 2.0.3 is uploaded to edgy and dapper-
<A> proposed; to test/use it on dapper, add an extra line to
<A> your /etc/apt/sources.list file:
<A>
<A> deb-src http://archive.ubuntu.com/ubuntu/ dapper-proposed
<A> main
<A> [...]
<A> Feedback is appreciated; please use the bug tracking system
<A> [...]

<B> Is there a chance of running this on Badger? Or there's a
<B> dozen of libs that are not/can not be backported to 5.10?"

<A> No, you need to build it on breezy. That can be done,
<A> but it requires some work; the libs are not the real
<A> problem (if you do look at [1] you'll see an icu backport
<A> as well), but reverting the renaming of the packages back
<A> to *2 requires some work. To do so, start with the 2.0.3
<A> packaging. The biggest grief will be the control file to
<A> get the replaces/conflicts correctly done114.
```

Les outils de communications habituels (*mailing-lists*, blogs, etc.) ne sont pas le seul lieu d'apprentissage technique, le principal étant à juste titre le code source, et ses dérivés puisqu'à travers le système de patchages et les systèmes de *versionning* du type bzt-NG, une confrontation directe au code source, code source commenté permet au lecteur de suivre le fil de pensée du programmeur, si ce dernier a fait l'effort de bien commenter son code, comme l'illustre l'extrait suivant tiré d'un patch d'un programme :

¹¹⁴ Source: *mailing-list* ubuntu-devel

```

+ recursiveXYC(N_page);
+ //by stopping to worry about creating empty nodes I made the
+ //code quite a bit more robust. This function makes sure we
+ //have a nice'n'clean tree
+ [...]
+ //I have blocks and columns, this function will turn that into
+ //paragraphs and columns
+ generateParagraphs();
+ [...]
+ //basically I first detect the text-alignment within blocks.
+ //ASSUMPTION: my block seperation thing is good enough so I
+ //don't need to worry about two alignments in one paragraph115

```

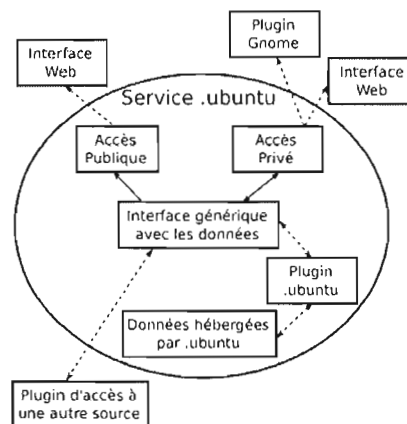
Il arrive fréquemment que les discussions deviennent techniques et portent sur les choix devant être effectués, comme l'illustre cette discussion sur le fonctionnement de Tribu, ce qui tend à éliminer de la prise de décision les personnes n'ayant pas les compétences techniques nécessaires, comme ici, la discussion prenant une allure assez technique que tous ne peuvent suivre :

```

<C> Je vais essayer de développer un peu plus l'idée d'utiliser
<C> Jabber/XMPP :
<C> - Jabber/XMPP est un protocole déjà existant, stable et
<C> robuste
<C> - Il peut servir à échanger à peu près n'importe quel type
<C> de données XML
<C> Et surtout :
<C> - Jabber/XMPP est extensible quasiment à l'infini par
<C> l'intermédiaire des JEP.
[... ]
<E> Je propose un schéma du type suivant :

```

¹¹⁵ Source: patch permettant au traitement de texte Abiword la conversion de fichiers PDFs
<http://www.jauco.nl/SoC/poppler-pdf-style-0.3.patch>



<E> Un service .ubuntu est composé :

- <E> - d'une interface générique permettant l'accès à des
- <E> données, par exemple : calendrier, mail, galerie photo, ...
- <E> - d'une interface web permettant une diffusion publique
- <E> - d'une ou plusieurs interfaces permettant la consultation
- <E> et la modification des données

<E>

<E> L'interface générique avec les données doit permettre de

<E> choisir un plugin pour accéder à des données. Le service

<E> .ubuntu proposera éventuellement son propre plugin quoi

<E> permettra le stockage des données en interne.

<E>

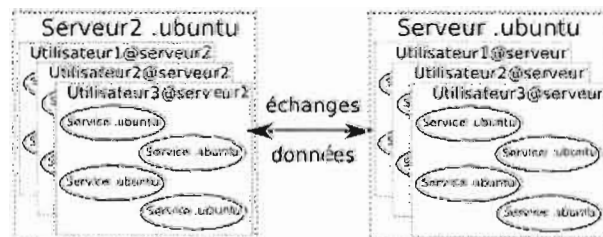
<E> Chaque utilisateur aura ses propres services. Les données

<E> peuvent être accessibles via un nom d'utilisateur et un nom

<E> de serveur. Comme ça on peut répartir les données sur

<E> plusieurs serveurs et effectuer des échanges, voir le schéma

<E> suivant.



[...]

<Q> [...] sinon on pourrait utiliser un schéma genre:

<Q> le logiciel est le daemon tribu qui tourne sur la machine et

<Q> gère l'intégration dans l'environnement.

<Q> _ le logiciel se connecte à tribu.org et m'identifie

<Q> _ tribu.org lui renvoie l'adresse du serveur sur lequel est

<Q> hébergé ma configuration ainsi que mon identifiant/mot de

<Q> passe

<Q> _ le logiciel se connecte à mon serveur personnel et

<Q> récupère toutes mes informations personnel

[...]

<E> [...] Par derrière on aura une API utilisable par les

<E> applications "clientes" (evolution, gthumb, ...) qui

<E> permettra d'accéder aux données des différents services
<E> indépendamment du fournisseur.

<N> Donc tu deviens une sorte de "super generateur" de
<N> configuration de l'outil de visualisation, c'est ça? le
<N> config.google ou config.flickr..
[...] ¹¹⁶

Les développeurs trouvent un plaisir certain dans le fait de coder, surtout lorsqu'ils sont « maîtres de leurs destins » (section 6.2.3). Le fait de mettre « *les mains dans le cambouis* »¹¹⁷, c'est-à-dire de coder, est particulièrement valorisé et le contributeur souhaitant s'améliorer reçoit généralement l'aide nécessaire. Cela a comme contrepartie un certain dédain pour les autres types de contribution comme le dénote ce message d'un des développeurs Ubuntu

Communications is all, but having a individual way to do things,
it's much more then a communication that fits us all. If you
want that, be a politician, but not a guy who is creating the
best and nicest Linux Distro ever¹¹⁸.

Cela donne une importance décisionnelle plus grande aux développeurs compétents, d'autant plus grande qu'ils contribuent beaucoup. Cette importance s'exerce par l'influence. Cela entraîne des prises de responsabilités volontaires de la part des contributeurs, qui acceptent de se contraindre à plus de travail sachant que leur influence en sera renforcée.

C'est pour pallier à la prédominance des développeurs sur les autres contributeurs qu'ont été établis dans la communauté Ubuntu les deux organes décisionnels (1) *Technical Board* se chargeant des décisions d'ordre technique, accessibles aux personnes étant reconnues pour avoir les compétences nécessaires et être impliquées dans la communauté, et (2) *Community Council*, regroupant toute sorte de

¹¹⁶ Fil de discussion de Tribu sur le forum Ubuntu-fr.

¹¹⁷ Source: Forum Ubuntu-fr

¹¹⁸ Source: Blog agrégé par Planet Ubuntu

contributeurs actifs (blogueurs, graphistes, etc.) participant aux choix de la communauté sur les aspects non techniques.

Dès lors que les choix se font sur des éléments techniques et qu'ils deviennent une source de pouvoir, il est nécessaire d'annoncer ces choix dans la communauté, comme dans le cas de la librairie de travail collaboratif obby, où développeurs annoncent et justifient publiquement leurs choix techniques :

The sequel to the current version (currently codenamed obby 0.5, but most likely this will change before a first release to prevent name conflicts with obby <=0.4.0) will be developed in C, not C++. This has several reasons:

- C++ has a quite huge impact on **code size**. The same program written in C makes the shared library and the executable much smaller. This might only apply to GCC and probably because it links half of the STL into each and every binary, but it hurts when I see that it is not necessary for a working library.

- **Binary compatibility** is easier to maintain. [...]

First of all, this means a complete rewrite of the library. This is a great opportunity to change the complete architecture. [...]¹¹⁹.

Pour tenir informés les contributeurs sur certains choix techniques, et ne pas avoir de déperdition de connaissances techniques, tant d'un point de vue fonctionnel que décisionnel la diffusion de message sur la *mailing-list* ubuntu-dev-announce peut se faire, comme dans ce cas :

```
Our current firmware locatin is
/lib/hotplug/firmware/$FIRMWARE-$VERSION
where $FIRMWARE is the firmware file name being requested, and
$VERSION is the kernel version.
```

Because of udev, we already have to move to the commonly used location of /lib/firmware. After discussion, we decided to ultimately move to this naming scheme:

```
/lib/firmware/$VERSION/$FIRMWARE
```

¹¹⁹ Source: « The future of (G)obby », <http://arbur.net/serendipity/index.php?url=archives/3-The-future-of-Gobby.html>

Packages known affected are the kernel (BenC) and linux-restricted-modules (infinity) and hotplug (Keybuk)¹²⁰.

Il demeure très important, pour garder le crédit de réputation et donc d'influence des développeurs de rester à jour dans leurs connaissances.

Tout ceci forme un système poussant les individus à l'entraide et forme un modèle d'innovations continues au sein de la communauté.

6.2.3 Une autogestion

Dans la distribution Ubuntu, les nouveautés sont discutées au sein de la communauté, dans les groupes *community council* et *technical board*. Toutefois, ces fonctionnalités suivent généralement un cycle bien précis pour être acceptées (voir Annexe 5) : il est nécessaire de rédiger la spécification en suivant un plan en huit étapes :

1. Résumé (Summary) ;
2. Argumentation (Rationale) ;
3. Étude de cas (Use cases) ;
4. Étendue d'utilisation (Scope) ;
5. Design ;
6. Implémentation ;
7. Résultats attendus (Outstanding issues) ;
8. Agenda et discussion (Bof agenda and discussion).

Cela correspond à la fiche descriptive de la spécification. Elle doit alors être enregistrée dans la partie gestion de projet de Launchpad, ce qui lui permettra d'obtenir un statut désignant son état de finalisation (en cours de discussion, accepté, etc.), sa priorité (faible, essentielle, etc.), son statut actuel (en cours de

¹²⁰ Message en date du 23 novembre sur la *mailing-list* ubuntu-devel-announce.

programmation, fini, etc.), les personnes s'occupant de coordonner cette fonctionnalité.

Si nous sommes ici dans le domaine de la gestion de projet classique, la particularité est que l'outil de gestion de projet a été développé (et est perpétuellement en cours de développement) par ses utilisateurs. Cet outil est conçu pour, mais aussi et surtout par les développeurs d'Ubuntu, correspondant alors fortement à leurs attentes. Ainsi, les développeurs discutent sur les *mailing-lists* aussi bien des améliorations de la distribution que du processus de conception de la distribution :

Hello everybody,

as much as I appreciate web2.0 coming to our part of town, I want to start a discussion about Tags in Malone. To get a good impression of what I'm talking about, see here:
<https://launchpad.net/distros/ubuntu/+bugs>

I want to propose a moderation queue for tags and get the current list completely cleaned up. Having a package tag (like 'firefox', 'epiphany' or 'gnome-panel') makes no sense at all. The same goes for 'gnome' and 'audio' - we have teams to handle this kind of bug.

It'd be nice if we could use the tags to organize our workflow. What about tags like 'go-upstream' or 'easyfix'? The current list is messy and doesn't make things easier. In my opinion we might just need a more prominent list of teams.

I propose to lock the 'add tag' UI and make it available to 'ubuntu-qa' or to whatever else team we can agree on.

I look forward to hear more views on this because I think tags are a tool that 'can' make our life easier.¹²¹

Les décisions sont prises par la communauté, sur *mailing-list* ou IRC :

<A> Hi Ubuntu developers,

<A>

<A> As discussed in Tuesday's Tech Board meeting, we need to
 <A> improve the workflow for sponsored uploads (or, rather,
 <A> instantiate such a process in the first place).

¹²¹ Source: *mailing-list* ubuntu-devel, l'emphase est de nous

```

<A>
<A> After some discussion we quickly agreed that it would be
<A> best to use Malone, Launchpad, and some magic scripts

<B> This looks fantastic! Improving this workflow will be very
<B> valuable for the developer community122.

```

Les personnes s'occupant de mettre en place les fonctionnalités sont les personnes motivées à les réaliser, qui acceptent de prendre des responsabilités plus grandes pour améliorer leurs compétences, leurs visibilités dans la communauté, leurs influences, ou pour le plaisir de contribuer, qui est toujours présent chez les développeurs. Il peut s'agir aussi de développeurs payés pour travailler sur la distribution, ces développeurs étant toutefois choisis selon leurs compétences et leurs réputations. Ils sont alors rétribués pour faire un travail qu'ils auraient probablement fait s'ils n'avaient été « que » contributeurs, la différence étant qu'ainsi, ils peuvent s'y consacrer à temps plein, à l'avantage de l'entreprise employeur, qui a (1) la possibilité d'orienter les trajectoires suivies par les projets de ses développeurs, (2) d'avoir une expertise en son sein sur des logiciels utilisés par ses clients, étant alors capable de répondre à leurs attentes en termes de correction de bugs ou d'ajouts de fonctionnalités.

La communauté codifie ses processus de fonctionnement à travers des contributions :

```
Hi Ubuntu developers
```

```
I have been working on a new document for the Ubuntu
Documentation on how to get involved with Ubuntu [1], and as
part of that, I have been going through all the team wiki pages
to gather basic information on how to get involved with each
team. Unfortunately, not all of this information is organized or
available in any easy way.
```

```
To solve this, I have made a TeamPageTemplate [2] in the Ubuntu
Wiki to offer a common structure for team wiki pages to make it
easy for new contributors to get an overview of the information
that they will need to get involved. I have already adapted the
DocumentationTeam wiki page to this template [3] so that you can
get an idea of what such a team wiki page can look like.
```

¹²² Source: *mailing-list* ubuntu-devel

I would like you to have a look at the ContributeToUbuntu section on Ubuntu Development [4] and writing new code for Ubuntu [5] so that you can make sure that it properly reflects how you want people to get involved. **Remember that this is to make it easy for new community members to get involved**, so please let me know if you have any input or comments on the suggested structure. [...]¹²³

Comme ce sont les développeurs et la communauté qui autogèrent leurs outils de gestion, certains logiciels sont ainsi conçus ou intégrés à la distribution dans le but même de simplifier le développement et la maintenance des logiciels. Ainsi, une des fonctionnalités nouvelles de Breezy, fut l'ajout dans les applications intégrées à la distribution d'un lien vers Rosetta, l'outil de traduction en ligne de Launchpad. D'autres logiciels, comme Bazaar-NG, ont pour objectifs de permettre la création de logiciel de façon plus décentralisée. L'une des nouveautés dans Edgy sera d'intégrer Xen¹²⁴ qui, outre un intérêt pour des machines en production (notamment des serveurs), permet aux programmeurs d'Ubuntu de tester de nouveaux paquets sans mettre en péril la stabilité de la machine en général. Cela permet ainsi des rapports de bugs simplifiés (pas de redémarrage ni de perte de données), de préserver son environnement de travail sur une zone « sécurisée », ...

D'autres fonctionnalités sont implantées, non pas pour améliorer directement les fonctionnalités pour l'utilisateur, mais pour permettre une meilleure correction de bug, ou un développement plus rapide et donc un meilleur gain de fonctionnalités dans le futur, c'est-à-dire finalement une plus grande efficacité :

with today's ubuntu-meta change (which added the 'apport-gtk' package to ubuntu-desktop), Edgy now has automatic crash reporting [1] [2]. It is not yet quite as automatic as it could be, since we do not yet have a proper bug reporting tool, but it should already help developers to track down crashes more efficiently:

¹²³ Source: *mailing-list* ubuntu-devel, l'emphase est de nous

¹²⁴ Voir <http://en.wikipedia.org/wiki/Xen>

* It intercepts crashes right when they happen the first time, which spares long sleepless nights for trying to reproduce the crash.

* The generated reports contain stack traces, a core dump, names and versions of the affected package and all of its dependencies, information about the OS, /proc, and runtime status. See the spec [1] for details. This should reduce the bug report ping-pong for common missing information.¹²⁵

Cette tendance à vouloir améliorer les outils n'est pas propre à Ubuntu, comme en témoigne l'existence du projet LDTP (Linux Desktop Testing Project) dont le but est :

producing high quality test automation framework and cutting-edge tools that can be used to test GNU/Linux Desktop and improve it¹²⁶.

Il y a dans le projet Ubuntu une volonté d'indépendance technologique sur les choix concernant le processus. Ainsi, la distribution ayant mis en place son propre système intégré de gestion, Launchpad, utilisant le logiciel Bzr-NG, l'outil de gestion de Bug Malone, etc. alors qu'il existe d'autres logiciels libres remplissant ces tâches comme bugzilla. Toutefois, pour un des contributeurs interviewés, Malone est beaucoup plus agréable à utiliser, et plus simple que ses concurrents. Cette volonté semble en effet prendre sa source dans le fait de vouloir rendre plus accessible les contributions à la communauté, ce qui résulte à la fois d'une plus grande efficacité du système, mais aussi d'un appel à plus de participation à la vie de la communauté donc une forme de démocratisation.

Tous les projets ne sont pas initiés par la communauté Ubuntu, mais le sont par des individus ou des groupes d'individus – formant des équipes de travail – pour être éventuellement intégrés plus tard. Ainsi, le projet Tribu a commencé suite à une discussion sur un forum de la communauté Ubuntu, et est maintenant autonome, même s'il sera peut-être intégré s'il aboutit à un résultat probant. De même, le projet EasyUbuntu dont le but est d'ajouter facilement des programmes utiles pour

¹²⁵ Source: *mailing-list* ubuntu-devel-announce

¹²⁶ Source: <http://ldtp.freedesktop.org/wiki/>

l'utilisateur à une distribution Ubuntu a été écrit par un individu et distribué sur son site internet, puis a été promu à travers planet et le forum dans la communauté locale francophone, et sera intégré à la prochaine distribution, Edgy.

Les équipes décidant d'intégrer le projet Ubuntu sont recensées dans Launchpad, qui, à la date du 18 août 2006, même si elles ne sont pas toutes actives, recense plus de 500 équipes de travail¹²⁷. Ces équipes fixent elles-mêmes leurs buts. Pour illustrer ceci, citons la Ubuntu-kernel-team, qui se charge d'adapter le noyau Linux, la « *MOTU Digital Photo team* » (voir section 6.2.4), une équipe ayant pour but de « *Release and keep up to date Digital Photo related packages* »¹²⁸, ou encore l'ubuntu-development-team, qui se définit ainsi :

This is the "Masters of the Universe", the general Ubuntu development team which has full upload rights to universe and multiverse¹²⁹.

Les rôles et buts des équipes sont donc variés puisqu'ils reposent sur des intérêts d'individus variés : il y a donc des équipes de développement, des équipes de traduction, ou encore des communautés locales.

Suivant la taille et le but des équipes, elles peuvent avoir des moyens de coordination propre, comme des *mailing-lists* dédiées, un site internet propre avec un espace sur le wiki. Dans le cas de la desktop team qui se définit comme

a regular Ubuntu team with meetings, with a mailing list, with bugs (oh yes, ...), an opinion how to do things and thrilling ideas how to let the Ubuntu Desktop evolve¹³⁰.

La communauté met à disposition de l'équipe pour réaliser ses objectifs :

- Our Mailing list:
<http://lists.ubuntu.com/mailman/listinfo/ubuntu-desktop>

¹²⁷ Source: <https://launchpad.net/people/+teamlist>

¹²⁸ Source: <https://wiki.ubuntu.com/MOTU/Teams/Photo>

¹²⁹ Source: <https://launchpad.net/people/ubuntu-dev>

¹³⁰ Source: <https://wiki.ubuntu.com/DesktopTeam>

- We'll collect visions and ideas on DesktopTeam/Visions
- Our short-term todo list: DesktopTeam/TODO
- We want to get new software in: DesktopTeam/NewSoftware
- Our news are in UDN: DesktopTeam/UDN¹³¹

Les équipes liées à Ubuntu ne sont pas contraintes d'utiliser l'infrastructure de Launchpad, comme dans le cas de l'équipe de Tribu, qui a débuté suite à une discussion sur le forum francophone et a préféré s'orienter vers SourceForge, même si elles sont encouragées à le faire pour participer à la vie communautaire.

Pour se coordonner, les équipes, sont généralement abonnées à la *mailing-list* ubuntu-dev-announce, qui diffuse des informations importantes pour les développeurs, tant d'annonces de réunions que des annonces des prises de décisions techniques importantes :

```
This was a lengthy discussion that actually touched on two
subjects. The quick summary is:
- Changed device names can cause failure to mount rootfs, and
thus failure to boot the system.
- The ProbeForRootFS goal addressed this for removable devices
(USB sticks and such) by mounting based on UUID, instead of
device path.
- This doesn't help the case such as bug 6367, where the change
is caused by docking station on a laptop (hda becomes hde).
- Widespread usage of UUID is not acceptable because of
limitations with ide-generic (do not want to load it without
reason, because it can break SATA and such), plus being so late
in dapper development.
So, several things were decided:
- ide-generic should not be loaded so soon. [...]
- To fix the original problem, <A> suggested that when the
initramfs is created, to store the UUID of the root fs. [...] 132
```

Elles sont aussi généralement abonnées à la *mailing-list* Ubuntu-dev, mais la fréquence des messages beaucoup plus élevée rend ceux-ci moins visibles.

Ces informations sont d'autant plus importantes, que certaines équipes travaillent en autonomie quasi totale, comme le révèle ce message :

¹³¹ idem

¹³² Source: *mailing-list* ubuntu-dev-announce

During the developer meeting today, mdz came up with the seemingly simple question: "do you have an idea when artwork will start to land in edgy?" I'm afraid the answer is going to be a drama in three acts, but without the drama, so I'll start explaining. [...] ¹³³

De ce fait, il arrive que des équipes différentes travaillent sur des problématiques similaires, mais avec des méthodes différentes, comme le laisse voir ce message :

```
<A> Hi to all,
<A> I'm Fabrizio from the UCK team, I'd like to tell you all
<A> that we've just released a new version of our customization
<A> kit.

<B> There is another very similar program being developed by
<B> other people named reconstructor. See
<B> http://reconstructor.aperantis.com/ for details. Maybe it
<B> would be wise to collaborate on this and create one program?

<A> that's really interesting, thank you very much for pointing
<A> it out, with uck team we'll try to find a way of
<A> collaborate134.
```

6.2.4 Des interactions décisives

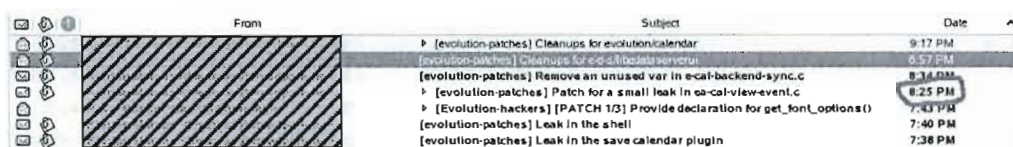
Ce n'est toutefois pas la majorité des équipes qui travaillent de façon indépendante. Un des avantages du logiciel libre est de pouvoir tirer profit du travail d'autres groupes. Ainsi, le projet Ubuntu intègre la dernière version de l'environnement GNOME dans sa distribution. Chaque version d'Ubuntu se voit gratifiée, en plus des ajouts faits à la distribution des nouvelles fonctionnalités et améliorations de chacune de ses composantes, et des composants de ses composantes, et ainsi de suite. Edgy bénéficiera donc des avancées de GNOME 2.16, par rapport à Dapper qui intègre GNOME 2.14. Ce message sur l'ajout d'une fonctionnalité dans Evolution, le logiciel de messagerie utilisé dans l'environnement GNOME profitera aussi par répercussion à GNOME 2.16 et donc Edgy :

¹³³ Source: *mailing-list* ubuntu-dev

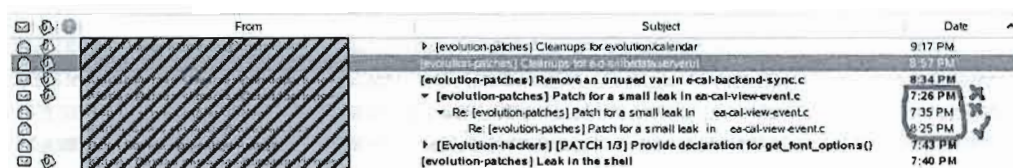
¹³⁴ Source: *mailing-list* ubuntu-dev

One big reason, I hate to use threading is that my reading is limited to the mails I received TODAY/RECENTLY. If a new message arrives in a old thread, it still lies way below the scrolled window and never comes to my visibility. I really like GMail at this very point and I tried to get something like this into Evolution, to improve threading usage. I got some good results.

If the message is in a collapsed state, the date of the THREAD, I compute to be the date of latest message received in that thread.



which brings the message on to the top to get the attention. If you expand, it appears like this.



If the message is in expanded state during sorting (List generation), then it takes the date of its creation, which puts it back to the original place. Once in a day, I can use View/Collapse All and see are there any new messages to a old thread.¹³⁵

Les équipes travaillent aussi conjointement. Par exemple, sur la *mailing-list* d'Ubuntu, un développeur GNOME vient ici donner une information importante pour l'amélioration des performances du thème graphique :

```
<A> I have been coding a small "torture" application for GTK
<A> themes [1], which mainly does two things: 1) crash test the
<A> engine by calling all the GTK paint functions with unusual
<A> parameters and 2) measure the time the theme takes to
<A> create / map / expose / resize / destroy widgets (with help
<A> from Federico's widget profiler's infrastructure). [...]
<A> Okay, so a few observations [...]
<A> Here, the details are interesting: Human is just a little
<A> slower than Clearlooks on most widget types (1.1 or 1.2
<A> ratios), but much slower on some particular ones:
```

¹³⁵ Source: Blog agrégé par Planet Gnome

<A> GtkProgressBar (5.5 times slower), GtkButton (5.5 as well),
 <A> GtkNotebook (2). I'm not very familiar with the Human
 <A> theme's code, but these appear to be the widgets that differ
 <A> the most (visually - glossy look) from Clearlooks, so it
 <A> probably won't be a surprise that they take more time to
 <A> draw (although 5 might be a bit higher than expected).

 I'm going to forward this data to the art team, this may
 be valuable information for deciding what to do about
 Human for Edgy (And the directions we will be taking it
 in)¹³⁶.

Donnons un autre exemple : l'intégration de « Slab », un logiciel issu de la distribution OpenSuse de Novell d'un projet nommé « *Better Desktop* » ayant pour but de fournir un environnement de bureau plus agréable et confortable pour l'utilisateur. À partir de Slab, une version a été modifiée pour Ubuntu, nommée USlab.

<A> <https://wiki.ubuntu.com/Slab>

<A>

<A> I hacked the source code myself...and it even works!

 That looks awesome!

<A> Thanks, USlab was specifically optimized for Ubuntu's needs
 <A> and requirements...

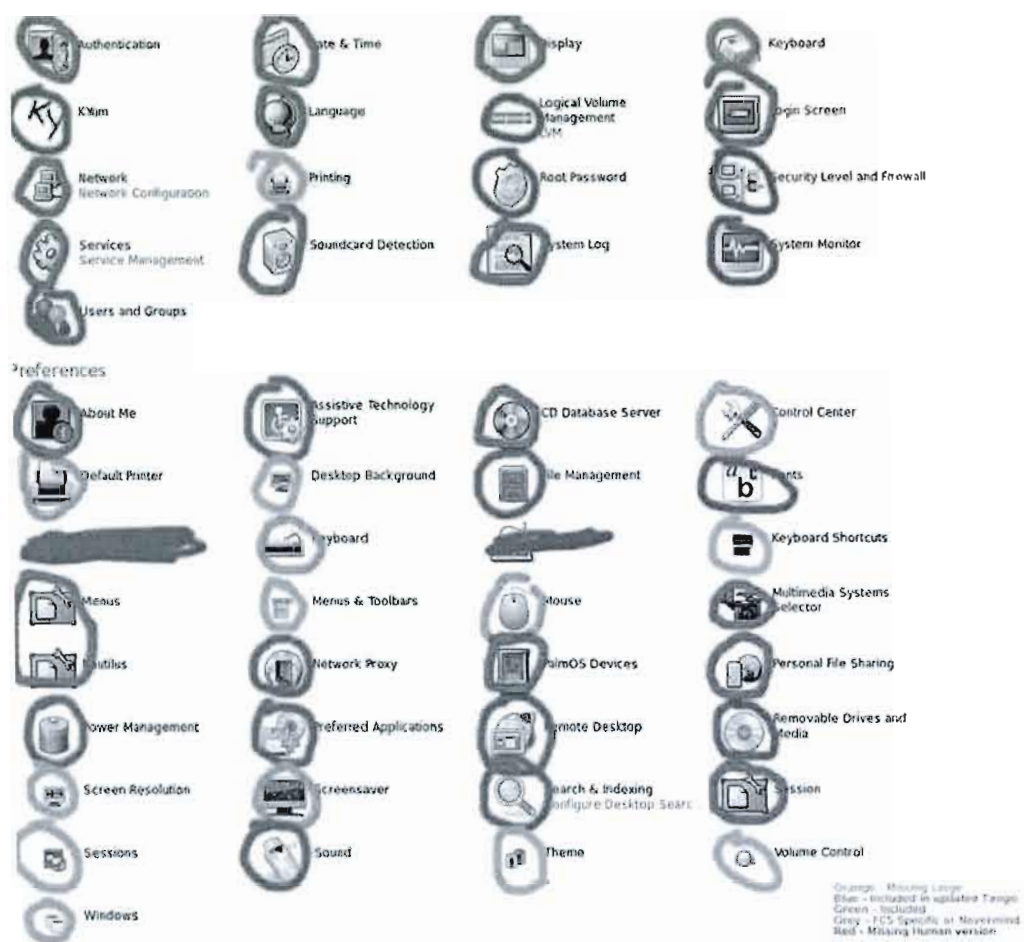
<A>

<A> I was referred to this list by someone on Who off ubuntu-
 <A> desktop (because he doesn't get much from ubuntu-desktop),
 <A> think you can all help me get my spec even closer to
 <A> possible inclusion?¹³⁷

USlab is a fork of SUSE Enterprise Desktop 10's new menu applet
 and application manager codenamed "Slab", but modified to fit in
 better in the Ubuntu environment.

¹³⁶ Source: *mailing-list* ubuntu-devel, l'emphase est de nous

¹³⁷ Source: *mailing-list* ubuntu-devel



(note, since I do use Fedora Core 5, some of the options in there might not actually be on Ubuntu or may have a Bluecurve icon)¹³⁸

Ainsi, il y a ici intégration du travail d'autres équipes créant aussi de la connaissance, comme GNOME, Network Manager, Beagle, Linux Kernel, OpenSuse, Slab, etc.¹³⁹

Illustrons mieux ce travail collaboratif avec cet exemple d'ajout de nouvelle fonctionnalité qu'est l'intégration d'un logiciel de création d'images panoramiques dans Ubuntu. Certains photographes amateurs et professionnels peuvent à l'aide d'un

¹³⁸ Source: <https://wiki.ubuntu.com/Slab>

¹³⁹ La liste est TRÈS longue

ordinateur et de plusieurs photos prises bout à bout créer une image panoramique. Cette fonctionnalité n'existe actuellement pas¹⁴⁰ dans la distribution Ubuntu.

La « *MOTU Digital Photo team* » est une équipe de la communauté s'étant fixée pour but de « *Release and keep up to date Digital Photo related packages* ». Parmi les plans pour parvenir à l'objectif annoncé, il y a celui de mettre en place un ensemble de logiciels qui, fonctionnant ensemble, permettraient de réaliser des images panoramiques de très grande qualité et ainsi permettre une nouvelle fonctionnalité dans la distribution : la création de panoramique¹⁴¹.

Les trois logiciels en question sont autopano-sift (détecte les points communs entre deux photos), Hugin (interface permettant la fusion des photos) et Enblend (supprimant les différences de contraste, de luminosité, et de couleurs aux frontières entre les photos fusionnées). Hugin est une interface graphique s'appuyant sur l'ensemble de logiciels 'Panorama Tools', logiciels qui par un ensemble complexe de fonctions mathématiques de traitement d'image sont à même de modifier des images pour en faire des panoramas de très haute qualité¹⁴².

L'un des objectifs fixés par l'équipe était donc d'intégrer ces logiciels à la distribution, ce qui a été fait pour autopano-sift. Toutefois, Hugin et Enblend ne l'ont pas été. En effet, Hugin et Enblend sont tous les deux diffusés sous licence GPL (licence libre) et utilisaient tous deux une librairie nommée 'VIGRA' (pour « *Vision with GeneRic Algorithms* »¹⁴³) qui était diffusée sous une licence « *VIGRA Artistic License* » non libre, donc non compatible avec les valeurs d'Ubuntu (et de Debian). Il

¹⁴⁰ En fait, cette fonctionnalité existe, mais elle n'a pas été intégrée dans Ubuntu, et nécessite un certain nombre de manipulations non triviales de la part de l'utilisateur, ce qui limite le nombre potentiel d'utilisateurs.

¹⁴¹ Source: <https://wiki.ubuntu.com/MOTU/Teams/Photo>

¹⁴² Source: <http://www.path.unimelb.edu.au/~dersch/>

¹⁴³ Source: <http://kogs-www.informatik.uni-hamburg.de/~koethe/vigra/>

était ainsi impossible d'intégrer VIGRA dans Ubuntu, et donc d'utiliser Hugin et Enblend.

Ainsi, étaient affectées non seulement la librairie VIGRA, mais par répercussion Hugin, Enblend et la chaîne complète de logiciels autopano-sift / Hugin / Enblend, ces programmes étant sans grand intérêt pris indépendamment, leur efficacité étant leur utilisation conjointe.

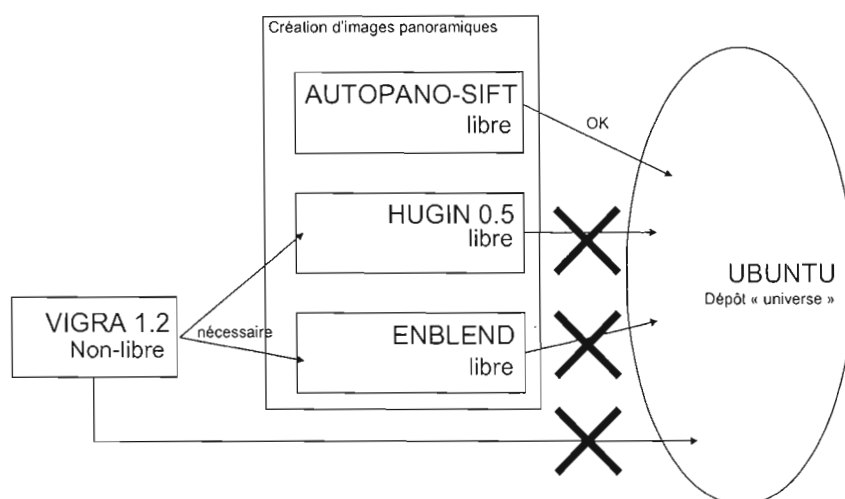


Figure 25 - Récapitulatif des liens entre logiciels

Pour remédier à la situation, deux démarches ont été entreprises. Tout d'abord en septembre 2005, <A>, mainteneur Debian de Hugin, a écrit sur la *mailing-list* de VIGRA pour informer les auteurs du problème de licence où il annonce que « *Vigra licence is not GPL-compatible* »¹⁴⁴. Il a été finalement choisi de changer la licence pour une version compatible GPL (licence MIT) « *we have discussed the licensing problem and came to the conclusion that we should place VIGRA under the MIT X11 License* »¹⁴⁵, chose qui a été fait effectivement avec VIGRA version 1.4 en décembre

¹⁴⁴ Source: *mailing list* VIGRA

¹⁴⁵ Source: *mailing list* VIGRA

2005 : « *Changes from Version 1.3.3 to 1.4.0 [...] Switched to the MIT X11 License* »¹⁴⁶.

Parallèlement, , mainteneur Ubuntu de Hugin et membre de la « Digital Team » apprend en janvier 2006 que Hugin ne pourra être accepté pour problème de licence :

```
08:00 <B>: gah, hugin rejected for license issues
08:00 <B>: VIGRA license
08:00 <B>: I'll have to study that more closely147.
```

Il crée alors un rapport de bug en février 2006, appelant les développeurs de Hugin à utiliser VIGRA 1.4 plutôt que la version 1.2 utilisée dans Hugin 0.5 :

```
Hugin source includes libvigra 1.2. The vigra license is
absolutely non-free and therefore non compatible with GPL that
hugin is released under. That is the reason why hugin has been
rejected from Ubuntu. Note that Enblend suffer from the same
issue.
Libvigra 1.4 is released under a MIT-style license. This
licensing is compatible with GPL and is considered Free.
```

What IMHO should be done:

```
-make hugin build with the system installed vigra 1.4
OR
-upgrade the vigra code in the source to version 1.4148.
```

Par ailleurs, il envoie un rapport de bug identique au développeur de Enblend, qui répond

```
Enblend uses a modified version of the vigra distributed with
Hugin, which in turn is a modified version of the official
vigra. Once Hugin updates to vigra 1.4, Enblend will be able to
follow suit. Perhaps all the patches will be incorporated into
the official vigra in the future.149
```

¹⁴⁶ Source: *mailing list* VIGRA

¹⁴⁷ Source: Canal IRC ubuntu-motu, le 20 janvier 2006

¹⁴⁸ Source:

http://sourceforge.net/tracker/index.php?func=detail&aid=1434981&group_id=77506&atid=550441

¹⁴⁹ Source:

http://sourceforge.net/tracker/index.php?func=detail&aid=1438069&group_id=123407&atid=696410

Cette information est alors rapportée sur le canal IRC Ubuntu-MOTU par le 22 février 2006 :

```
01:16 <B>: hugin will wait because of the license
01:16 <B>: but they'll fix the problem upstream
01:16 <B>: by using vigra 1.4 that is MIT licensed
```

Les développeurs de Hugin quant à eux, prennent en compte la demande, et Hugin 0.6 sort le 23 juillet 2006 et utilise VIGRA 1.4, donc n'est dépendant d'aucune librairie sous une licence non libre :

```
Newsworthy changes in 0.6 [...] updated vigra to version 1.4.
All sourcecode of hugin is now released under approved open
source licences. This means there is no problem with including
hugin in debian and ubuntu anymore150.
```

Le 26 juillet 2006, propose d'inclure Hugin 0.6 à Ubuntu, qui est accepté le 3 août suite à la correction d'un bug le 28 juillet 2006 du package Ubuntu :

```
<B> [26 juillet 2006] : This is a new upstream version. The
<B> previous one had licensing problem with libvigra. The newer
<B> version of vigra is licensed under a MIT-style license. It
<B> is the one that hugin 0.6 uses. [...]

<C> [27 juillet 2006] : UniversalCursor.cpp:69:21: error:
<C> gtk/gtk.h: No such file or directory [...]

<B> [28 juillet 2006] : added an explicit libgtk2.0-dev build
<B> dependency to not FTBS.151
```

L'information est alors relayée par le 24 juillet 2006 sur son blog, repris par l'agrégateur planet ubuntu, pour informer les utilisateurs de l'arrivée prochaine dans la distribution du logiciel :

```
Hugin got rejected because of libvigra. Fortunately this problem
will be addressed upstream. Same would happen to enblend (I have
to report the problem upstream)152.
```

Et reprise sur le planet francophone le 27 juillet 2006 :

¹⁵⁰ Source: http://sourceforge.net/forum/forum.php?forum_id=593941

¹⁵¹ Source: <http://revu.tauware.de/details.py?upid=2797>

¹⁵² Source: Blog de

La nouvelle version (0.6) de hugin vient de sortir. La demande est déjà faite afin que ce superbe logiciel permettant de réaliser des panoramas soit enfin intégré dans la Ubuntu. Ce n'est que depuis la 0.6 que la bibliothèque vigra 1.4 est utilisée. Avec cette dernière (* Switched to the MIT X11 License.) est levé le blocage qui existait sur la licence de hugin qui n'était pas compatible avec notre distribution¹⁵³.

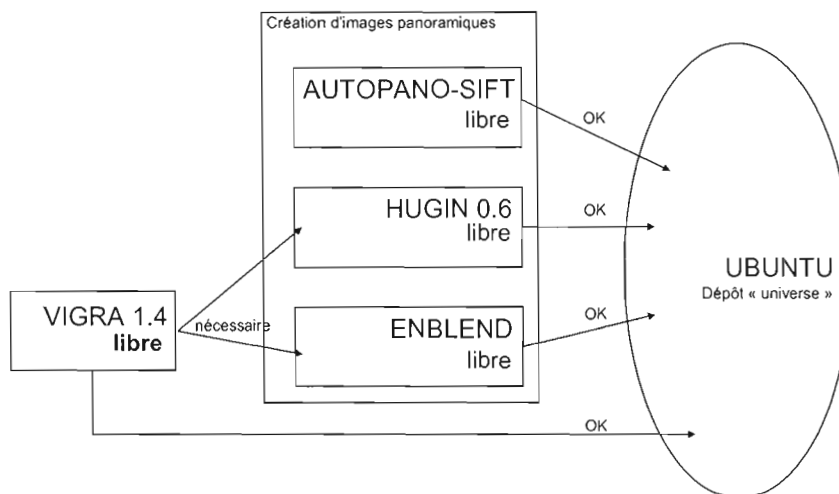


Figure 26 - Intégration des logiciels après le changement de licence, et les modifications

À partir de cet exemple servant d'illustration à des phénomènes très courants, voir banals dans la communauté du logiciel libre, nous pouvons représenter les différents groupes de travail interagissant de façon dynamique et chaotique au sein de la communauté tel que cela est présenté sur la Figure 27.

¹⁵³ Source: Blog agrégé par Planet Ubuntu-fr

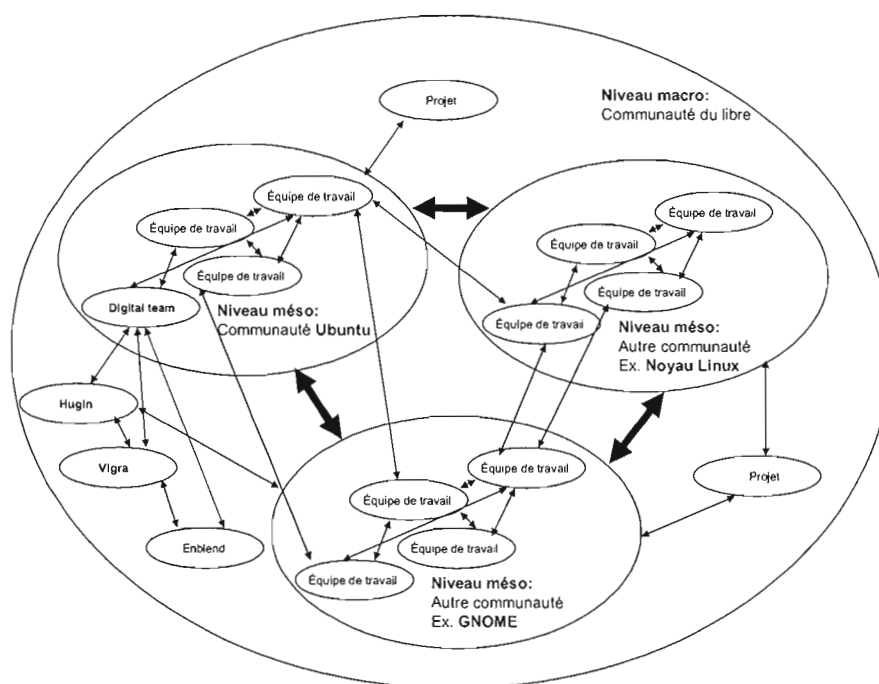


Figure 27 - Illustrations du travail dans la communauté du logiciel libre

6.3 De nouvelles connaissances

Dans une distribution telle qu'Ubuntu, les artefacts, c'est-à-dire les logiciels, les fonctionnalités, les rapports de bugs, la documentation, etc. représentent des artefacts résultant d'un travail ou d'une volonté de codification, voire pour le logiciel en lui-même de codage au sens d'action d'écrire du code.

La forme ultime d'artefact représente la distribution elle-même. Le chemin vers lequel s'orientent les nouvelles connaissances est différent suivant les versions. Pour la version Dapper d'Ubuntu sortie en juin 2006, la communauté s'était fixée pour objectif d'en faire la version la plus stable possible, la plus agréable possible pour l'utilisateur. L'accent n'a pas été mis sur l'intégration de nouvelles fonctionnalités innovantes, mais sur une convivialité, une intégration créant un liant sensible entre l'utilisateur et son système informatique.

La version 6.06 d'Ubuntu est cependant allée plus loin en produisant un mélange rare entre plusieurs éléments clés. La distribution se présente sous la forme d'un Live CD qui sert également d'installateur, ce qui facilite la mise en place d'un système, tout en permettant de le tester sans danger. Mais c'est surtout l'association de deux notions qui marque Ubuntu : la fiabilité et les performances. La petite distribution ne manque ni de l'une, ni de l'autre, au point qu'elle est devenue très rapidement une alternative crédible dans le monde professionnelle¹⁵⁴.

Cette volonté d'aboutir à une version stabilisée et *user friendly* a tout de même été l'occasion de nouveaux ajouts de fonctionnalités, comme un nouvel installateur permettant d'effectuer une installation à partir d'un CD *bootable*, c'est-à-dire que le CD permet à l'ordinateur de démarrer et d'utiliser un ensemble de programmes sans avoir besoin de disque dur, ce qui a nécessité beaucoup de travail en terme de conception de la part des contributeurs, ainsi qu'un important travail de débogage.

Avec Edgy, la communauté voulait entamer un nouveau cycle pour aller vers une nouvelle génération de distribution, ceci laissant alors place à un grand nombre de nouvelles fonctionnalités innovantes, quitte à perdre – temporairement – en convivialité et en stabilité.

Edgy. C'est plus qu'une nouvelle version, c'est un nouveau départ pour Ubuntu. Le premier a abouti à Dapper, qui devrait avoir un très bel avenir. Le suivant pourrait être encore plus merveilleux et Edgy qu'un début¹⁵⁵.

Aussi, les nouveautés prévues pour cette version sont importantes, et d'autant plus que la liste des nouveautés est en perpétuel changement, et que cela motive beaucoup de contributeurs.

Nouveautés techniques

Afin de séduire de plus en plus d'adeptes, *Edgy Eft* mettra l'accent sur l'intégration de la technologie de virtualisation **Xen** (logiciel libre hyperviseur qui permet de faire fonctionner plusieurs OS virtuels sur une machine hôte), ainsi que sur

¹⁵⁴ Source: Ubuntu, la distribution qui monte sans cesse, PC Impact

¹⁵⁵ Source: Blog traitant d'Ubuntu

celle du nouveau serveur graphique 3D **Xgl** (surcouche de OpenGL) ou **AIGLX** (choix à définir). Egalement à venir, un nouveau gestionnaire de réseau, le réel support de multi-archives 32 bits et 64 bits en vue de mieux exploiter les possibilités des machines à base d' AMD 64 ou encore l'intégration du nouveau gestionnaire de paquetages **SMART**.

Shuttleworth reste toutefois ouvert à toute autre idée pour agrémenter son OS et exhorte la communauté des développeurs du monde du libre (geeks compris) à lui en faire part pour comme il le dit : " rêver un peu "¹⁵⁶.

Il y a ainsi une forme de boucle de rétroaction positive entre motivation des contributeurs et création de connaissance.

Ceci représente un grand nombre de nouveautés. L'intégration de Xen permet de virtualiser des systèmes d'exploitation, c'est-à-dire de les lancer tels des logiciels, rendant alors leur développement plus simple et étant aussi très utile pour l'administration de serveurs. XGL permet d'obtenir un bureau à trois dimensions avec N bureaux virtuels présents sur N faces d'un polyèdre, et ajoutant de nombreux effets visuels esthétiques et fonctionnels. SMART est un tout nouveau système de gestion de paquetage permettant une gestion plus fine de la maintenance du système de la part de l'administrateur. Edgy ajoute des fonctionnalités innovantes comme :

[...] **GnomeAppInstallPopconSpec** : Suggérer l'installation de packages sur la base de ceux installés par les autres utilisateurs¹⁵⁷.

Outre les améliorations techniques, un travail important de documentation est effectué, à travers un wiki. Ainsi, en compilant et ordonnant les « *pages les plus utiles du site* »¹⁵⁸ de documentation, l'équipe locale francophone a créé un guide de référence d'environ 200 pages.

¹⁵⁶ Source: « *Edgy Eft ou la mouture automnale d' Ubuntu* », Génération NT

¹⁵⁷ idem

¹⁵⁸ Source: Blog agrégé par Planet Ubuntu-fr

Toutes ces améliorations représentent des artefacts bien visibles. La distribution s'améliore continuellement, car les contributeurs apprennent et font émerger de la connaissance tacite dont le résultat « mesurable » est la connaissance codifiée sous les formes précédemment évoquées. Les nouvelles versions représentent une forme de flux cumulés d'accumulations de nouvelles connaissances.

Il ressort qu'au fur et à mesure de l'intégration et du travail effectué, le choix de ce que sera la distribution, de la trajectoire technologique qu'elle va suivre s'affine, celle-ci émergeant avec la construction de la distribution.

6.4 De la(dis)continuité dans le temps

À chaque nouvelle distribution, le cœur des développeurs fixe un objectif général, celui-ci orientant la création dans la distribution jusqu'à la prochaine version (voir section 6.1). Pour Dapper, il s'agissait de faire une version la plus stable et agréable possible dans la continuité des précédentes versions. Pour Edgy, il s'agit de recommencer un cycle d'innovation et de création, avec l'introduction de nouvelles fonctionnalités, quitte à ce qu'elles soient moins bien adaptées, mais qu'elles permettent un nouvel usage de l'informatique. L'objectif directeur est discuté en commun par les parties prenantes, c'est-à-dire le *technical board*, le *community council*, mais pas à travers toute la communauté.

Il a été décidé dans la communauté de faire évoluer Dapper vers une distribution la plus accessible possible, afin de 'capitaliser' tout le travail fait sur les précédentes versions :

We're likely to want to be a bit more conservative in the feature goals, and freeze a little longer. [...] The 6.04 release will be the culmination of 2 years hard work, with Warty, Hoary and Breezy having set the style and the pace. We want it to represent the fruit of that labour - it brings to a close this first major cycle of development for Ubuntu. It will be a release that people can plan for, participate in and build solutions on top of. It will get even more polish than we put

into Breezy, so we think it deserves a name that reflects that polish¹⁵⁹.

A posteriori, face à la déception de certains membres, le terme "polish" a été jugé trop vague.

To "polish" a release means to fix lots of little things that are irritating but not essential, and of course, everyone has their own list of bugbears. So saying the release will be "more polished" is about as bad as saying it will be "more better" :D. In future, I think we should articulate specifically where we will be investing the effort. In the case of Dapper, it was (a) the kernel, (b) the installer, (c) the server, and (d) the graphics theme and icons. Of course there was a ton of other work done, but that was what *I* meant by "polish". So we need to communicate much more clearly EXACTLY where the investment and the results will be found¹⁶⁰.

Avec Edgy, il s'agit de redémarrer le cycle d'introduction important de nouvelles fonctionnalités, qui bien que s'appuyant sur tout le travail précédemment effectué, marque une discontinuité dans l'approche générale de ce que doit être distribution :

And here's why. Edgy is all about cutting edge, perhaps bleeding edge, brand new code and infrastructure. It will be the right time to bring in some seriously interesting but definitely edgy new technologies which lay the groundwork for the next wave of Ubuntu development.

[...]

We can afford to take some risks with Dapper+1, because Dapper has turned out so well. We have a great answer for people who need super-solid and super-predictable results: Dapper is still fresh, will continue to work on modern hardware for some time, and has plenty of legs in its support cycle left to run.

In terms of the management of the release, we will have some fun with the core Canonical team. I'm promising to impose (almost ; -) zero from-the-top requirements for Edgy, this release is entirely up the to development team to envision and implement. Of course, feature proposals need to go through a review and approval process, and we'll make sure everyone has enough on their plate to keep busy during the cycle, but almost everything that lands in Edgy will be driven from the development team, who get to play with whatever new technologies they fancy along the

¹⁵⁹ "The name of Breezy+1", Mark Shuttleworth sur la *mailing-list* ubuntu-dev-announce - <https://lists.ubuntu.com/archives/ubuntu-announce/2005-September/000034.html>

¹⁶⁰ Source: <http://www.markshuttleworth.com/archives/50>

way. So that should give us a nice big bump in infrastructure and bling.

I would encourage members of the community who have been thinking of a cool new feature or plan to seize the opportunity to get it into Edgy.

[...]

I think a clean-the-pipes type release like Edgy is a good way for us to re-energise the team and the distro. Risk is good, when you give it a place and a time. And Dapper+1 is the right time for us to take a few risks¹⁶¹.

Cette notion de cycle pour la distribution est importante et partagée

Dapper is not supposed to be as stable as Debian unstable in this phase of the development cycle¹⁶².

Dans un même cycle, il y a évolution dans la continuité, Dapper s'appuyant sur Breezy, s'appuyant sur Hoary, s'appuyant sur Warty toujours dans le même courant, dans la même vision de ce que doit être la distribution. Toutefois, au-delà du cycle d'une distribution, l'évolution des différentes versions des distributions se fait par une (r)évolution conceptuelle marquée de discontinuité dans le chemin ou la trajectoire technologique qu'elle va suivre, bien que s'appuyant évidemment quand même sur les efforts et les améliorations des précédentes versions. C'est ce qui est présenté dans la communauté comme le métacycle :

This "meta cycle" of aggressive new features, slowly converging over a series of releases on a solid and consistent look-and-feel and underlying platform, has worked very well for us over the course of the past two years. We didn't plan it this way, but I suspect the next two to three years will look similar - we'll start of with a release that has a lot of edge and new tech (remember Warty?) and polish that up till we see the timing is right for a really polished enterprise "long term support" release, like Dapper. We've no concrete plans for the next Dapper, only that we'll know a release or two in advance when the time is right¹⁶³.

¹⁶¹ "Planning Dapper+1", Mark Shuttleworth sur la *mailing-list* ubuntu-dev-announce, <https://lists.ubuntu.com/archives/ubuntu-announce/2006-April/000064.html>

¹⁶² Source: *mailing-list* ubuntu-dev

¹⁶³ "Planning Dapper+1", Mark Shuttleworth sur la *mailing-list* ubuntu-dev-announce, <https://lists.ubuntu.com/archives/ubuntu-announce/2006-April/000064.html>

Nous interprétons cela comme des trajectoires technologiques suivies résultant des choix collectifs et de l'environnement forgeant une piste, mais limitant les possibles au cours d'un cycle, d'où l'importance des métacycles, rouvrant des sentiers technologiques précédemment fermés, jusqu'aux prochains enracinements dans les futurs choix émergents de la communauté et de l'environnement.

Les valeurs d'Ubuntu (section 6.1.2) ont une incidence sur les choix et les objectifs déterminés. Ainsi, pour Edgy, certaines fonctionnalités nouvelles concernent l'accessibilité aux personnes handicapées, ou encore une simplicité plus importante pour les membres de la communauté, afin de l'élargir :

Accessibilité : Intégration d'un lecteur audio d'écran, un clavier sur l'écran, un boot parlant et autres programmes pour aider les personnes avec un handicap (Ubuntu, Kubuntu, Xubuntu), mais aussi mise en place d'un site web récapitulant toutes ses améliorations. [...]

Communauté : Amélioration de l'animation autour des dépôts Universe (faciliter l'accès, l'intégration ...), intégration de popcon (pour savoir quelques packages sont populaires), documentation des changements dans les paquets, liste des paquets officiellement supportés¹⁶⁴.

Ces cycles d'évolution ont des périodes de temps déterminées, se fixant au rythme des distributions, c'est-à-dire avec une échéance d'environ six mois entre leurs définitions et leurs accomplissements.

Certains objectifs sont trop ambitieux pour être réalisés dans ce laps de temps et ne sont pas intégrés au cycle courant, au profit d'un autre cycle à venir, voire sans date d'échéance s'il n'est pas possible d'en fixer. S'ils ont été mal évalués en terme de temps nécessaire, les objectifs sont repoussés à une distribution ultérieure, comme c'est le cas dans Edgy avec l'intégration de SMART et XGL :

¹⁶⁴ Source: Blog traitant d'Ubuntu

Depuis son annonce Edgy Eft est désignée comme une version contrastant avec Dapper Drake qualifié de fiable voir très fiable. Mark Shuttleworth avait alors demandé aux développeurs de fixer des objectifs ambitieux pour cette version afin de la démarquer de ses concurrentes.

Oui mais voilà je ne suis plus le seul à me demander à la lecture des Specifications d'Edgy si finalement elle n'aurait « d'edgy » que la phase de développement plus courte que prévu (en effet 6 semaines de développement ont été réattribuées à Dapper).

Néanmoins ne nous emballons pas cette version peut réserver encore bien des surprises notamment par l'apport des nouvelles versions de tout les packages depuis Debian, ou encore dans la future version de GNOME et de KDE.¹⁶⁵

Les valeurs pragmatiques de la communauté tendent à renforcer les projets faisables dans des temps raisonnables, ou assez rapidement utilisables, même partiellement, sur l'élaboration d'un projet ambitieux mais sans aucune possibilité – à court et moyen terme – d'utilisation.

Au cours de ce chapitre, nous avons présenté nos résultats, présentés sous la forme de trois catégories, l'engagement à la communauté, le fonctionnement mérito-démocratique et la (dis)continuité dans le temps, toutes trois permettant la création de connaissance à travers les processus de travail de la distribution. Nous allons dans le chapitre suivant analyser ces résultats.

¹⁶⁵ Source: Blog agrégé par Planet Ubuntu-fr

CHAPITRE VII

ANALYSE DES RÉSULTATS

Dans ce chapitre, nous allons confronter les résultats présentés au chapitre précédent avec le cadre conceptuel élaboré au chapitre 3. Pour ce faire, nous commencerons par la description d'un modèle de création de connaissance adapté à la communauté Ubuntu, puis nous présenterons les conditions d'activation de connaissances décrites dans la littérature retrouvées sur le terrain, et enfin, à travers la validation du modèle SECI nous présenterons les conditions d'activations particulières à la communauté. Nous exposerons par la suite les limites de la recherche ainsi que les perspectives d'avenir.

7.1 Une description d'un modèle de création de connaissance à l'aide du modèle de la firme créatrice de connaissance

À l'issue de notre recherche, il ressort que le modèle de la firme créatrice de connaissance est approprié par rapport aux résultats émergents de la recherche menée dans la communauté Ubuntu.

Ainsi, les valeurs partagées dans la communauté (ennemi commun, humanisme, liberté, bidouille, partage d'information, autodérision, démocratisation) permettent, en s'appuyant sur ces valeurs, de déterminer une vision du futur de ce que doit être l'informatique. Cette vision, qui dans le cas d'Ubuntu est traduite par « *linux for human beings* », s'appuie sur deux idées clés à savoir « *linux* » faisant référence aux valeurs et systèmes *hackers*, et « *human being* », l'être humain. Cette vision est alors partagée par la communauté et répond effectivement à la question « *pourquoi faisons-nous ce que nous faisons* », et est créatrice de sens pour les acteurs de la communauté.

Nous avons aussi été amenés à voir que cela était traduit dans des cycles et des métacycles de création, formant des objectifs alternants continuité et rupture, continuité pour l'amélioration et l'aboutissement, rupture pour permettre la réorientation, le changement au sein de la communauté. Si les objectifs d'un métacycle émergent au fur et à mesure de l'évolution technique, les objectifs énoncés sur la période d'un cycle permettent de rendre la vision concrète, les acteurs sachant alors dans quel sens créer. Ces objectifs s'appuient sur la vision et les valeurs de la communauté comme cela a été présenté section 6.1.1.

Le fonctionnement démocratique – nous l'avons vu section 6.2.1 – engendre la liberté de parole pour tous. À travers les nombreux échanges, parfois même houleux, émergent des fils de discussions des connaissances nouvelles. Si le fil de discussions peut paraître parfois futile pour le profane, précisons que bien souvent, il ne l'est qu'en apparence et joue un rôle important de catalyseur de connaissances, tant pour les participants actifs que pour celui qui suit la discussion, car elle est – comme beaucoup de discussions débutant sur des trolls et ne dégénéralant pas (ou ne dégénéralant plus s'ils sont recadrés) – instructive et permet (1) au participant de créer des connaissances, en opposant des avis divergents et (2) à toute personne participant ou relisant le fil de discussion, en suivant le processus de construction, d'activer des connaissances grâce à un transfert reconstruisant le fil d'émergence de la connaissance, et ainsi d'apprendre par re-expérimentation, tel que l'ont montré Hemetsberger et Reinhard (cf. section 2.6).

L'une des particularités de la communauté libre est la possibilité donnée à l'utilisateur de devenir acteur, participant en tant que contributeur au logiciel qu'il utilise. Dans la communauté Ubuntu, la volonté de faire participer l'utilisateur est forte, cela passe par une participation possible à différents niveaux (traduction, artistique, code). Nous avons vu dans la communauté l'importance donnée au code, aux pratiques et à la noblesse de la chose technique. Ceci correspond à ce que Nonaka

et Toyama qualifient de pratique, et permet ainsi de résonner et de faire un contre poids synthétique face au dialogue, pratique et dialogue fonctionnant en dialectique, ou pour dire la même chose dans un vocabulaire cybernétique, dialogue et pratique forment une boucle à rétroaction positive en termes de nouvelles créations.

Cette activité de dialogue et de pratique s'effectue au sein d'équipes de travail autogérées. Dans la littérature, ces équipes de travail propre au logiciel libre sont généralement qualifiées de communautés de pratique ou de communautés épistémiques, ce qui correspond effectivement aux équipes de travail du terrain. Elles forment aussi des « Ba » selon le concept de Nonaka et Konno. Les « Ba » sont définis par Nonaka et Toyama section 1.3.4 comme étant « *as a shared context in motion, since it is moving. Through interactions with others and the environment, both the contexts of ba and the participants grow. New knowledge is created through such changes in meanings and contexts* ». Si nous préférons analyser ces équipes de travail autogérées sous la forme de « Ba », plutôt que sous la forme de communautés de pratique, c'est que, comme le note Nonaka et Toyama, les trois principales différences (cf. section 1.5.2) sont que :

4. la communauté de pratique est le lieu où les membres apprennent la connaissance qui est dans la communauté, alors que le « Ba » est un lieu de création nouvelle de connaissance;
5. la communauté de pratique a une identité, une frontière, une culture, une histoire alors que le « Ba » a des frontières changeantes et fluides, et peut changer rapidement, ainsi que ses participants;
6. les membres d'une communauté de pratique sont assez stables, cela prend un certain temps aux nouveaux participants pour être pleinement intégrés, alors que les membres d'un « Ba » ne sont pas fixés, ils vont et viennent.

Ces trois conditions étant respectées puisque dans le cas présent, les équipes de travail, créent de la connaissance, présentent une importante dynamique externe et interne. Nous avons vu la multiplicité des « Ba » dans une communauté telle qu'Ubuntu, et d'autres communautés du libre (les frontières entre communautés sont loin d'être nettes) se traduisant par de nombreuses interactions. Pour faciliter les « Ba » au sein de la distribution, les développeurs d'Ubuntu cherchent à décentraliser au maximum les prises de décisions aussi bien en promouvant cette valeur qu'en la rendant possible grâce à des outils techniques comme Launchpad, Bzr-NG (cf. section 5.2.8), ce qui en fait la composante autogérée de ces « Ba ». Ces groupes se rassemblent fréquemment pour former un « Basho », c'est-à-dire un ensemble de « Ba », résultant de leurs interactions, comme nous l'avons montré dans le cas de l'intégration du logiciel Hugin au sein d'Ubuntu.

Nous avons aussi présenté la diversité des membres de la communauté. Ces derniers viennent de pays différents avec des connaissances différentes, des approches différentes, mais souhaitent travailler tout de même ensemble au sein de la communauté. Cette diversité forme un écosystème de connaissances, toujours fluctuant, dans lequel « nage » la communauté Ubuntu.

Enfin, nous avons eu l'occasion de présenter ce en quoi Ubuntu créait de la connaissance dans la section 6.3, cette connaissance étant large tant en tant qu'outil informatique à proprement parler (logiciel d'installation par exemple), qu'à travers Internet, le wiki rassemblant documentation, expériences, trucs et astuces, commentaires, les réponses de la communauté aux questions à travers les *mailing-lists* ou l'usage de forums, ou encore directement la discussion sur IRC avec d'autres utilisateurs sous forme d'artefacts, mais aussi enracinés dans les individus. C'est ce que Nonaka et Toyama appellent dans le modèle de la firme créatrice de connaissance l'actif de connaissance, qui représente la connaissance émergente de l'interaction entre le dialogue et la pratique.

Nous pouvons dès lors construire le modèle de fonctionnement de la communauté libre Ubuntu présenté en Figure 28.

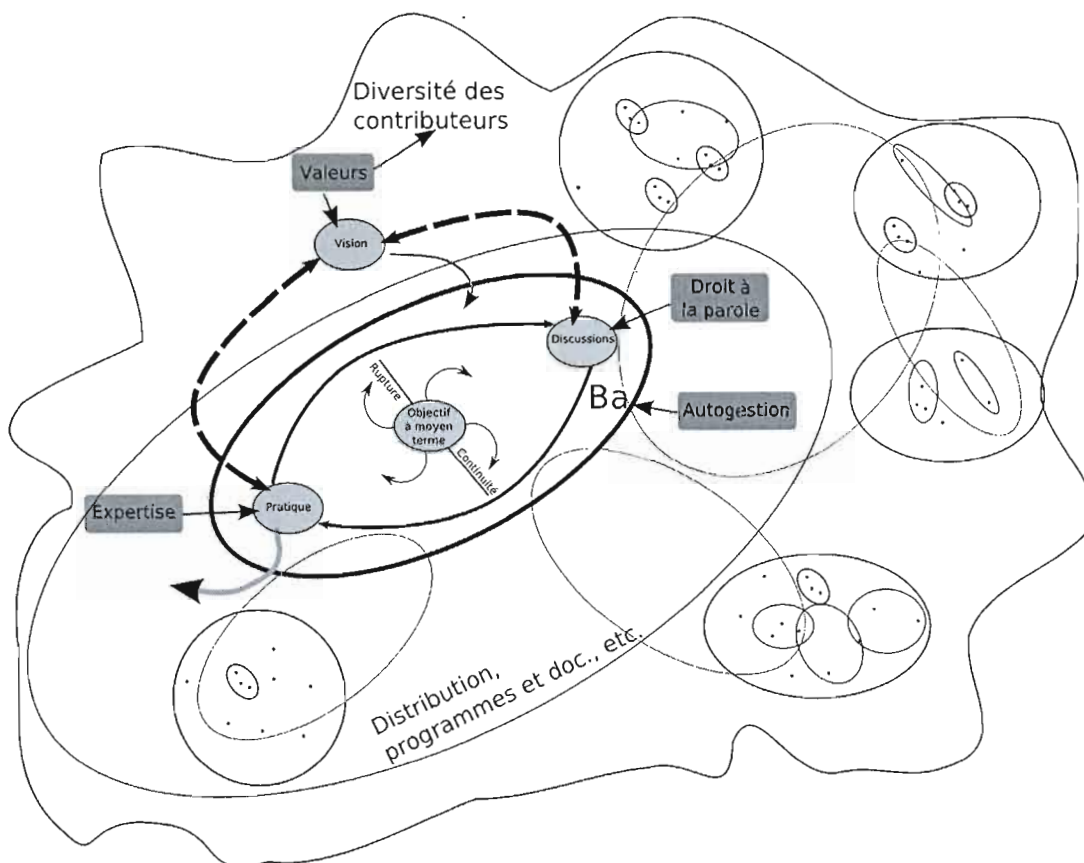


Figure 28 - Présentation d'Ubuntu comme firme créatrice de connaissance

7.2 Les conditions d'activation de Nonaka présentes

Nous mettons ici en lumière les conditions énoncées par Nonaka et Takeuchi (section 1.3.2.2) comme étant les conditions de création de connaissance.

L'intention principale de la distribution Ubuntu est de faire du système un système pour les êtres humains, et plus concrètement, pour chaque version de la distribution, un certain nombre d'objectifs sont affichés, il y a ici concordance entre la vision et

l'intention de la communauté. Celle-ci permet de se former une vision claire de l'avenir, afin que tous les membres de la communauté puissent travailler ensemble sur des bases communes. Ainsi, même s'il y a des désaccords au sein d'une équipe de travail, celle-ci ne se disloque pas, car elle reste soudée par cette finalité partagée et cherche souvent à trouver une façon originale de concilier ce qui auparavant ne l'était pas.

L'autonomie est présente partout au sein des différentes équipes de travail, celles-ci étant autogérées. Ubuntu fournit des infrastructures de travail aux équipes, mais celles-ci ne sont pas imposées. Elles sont toutefois souvent utilisées, car elles permettent une plus grande efficacité pour arriver aux objectifs communs des différentes équipes, et permettent par ailleurs de faire « vivre » la communauté en lui fournissant des lieux d'échange. Cette autonomie permet aux contributeurs de travailler selon leurs propres contraintes, et selon leurs propres conceptions du travail et de l'informatique. Dans une communauté aussi variée qu'Ubuntu, cette autonomie est une condition essentielle, quand on pense par exemple au décalage horaire des individus répartis sur tous les fuseaux, qui ne pourraient travailler ensemble si des contraintes de temps étaient fixées. De plus, en augmentant la diversité des méthodes de travail – chacune disposant d'avantages et d'inconvénients divers — on cumule les avantages pour la distribution, les inconvénients ayant quant à eux tendance à se compenser. Ainsi, chacun peut se fixer ses propres contraintes dans la mesure où elles sont compatibles avec sa conception du travail.

La fluctuation et le chaos sont présents partout dans la communauté. Ils sont même valorisés, comme le reflète une acétate de présentation du process d'Ubuntu lors de la réunion Ubuntu Below Zero en octobre 2005, qui précisait : « *What we expect : some chaos* ». Il y a toutefois la volonté d'en faire un chaos créatif avec la mise en place d'infrastructures ordonnant, ou permettant un minimum aux contributeurs de se retrouver dans la masse d'information présente. Ce chaos est un élément

particulièrement important, car il permet de mettre en contact des personnes ou des idées qui n'auraient normalement jamais dû se rencontrer si elles avaient été bien étanchement classifiées. La contrepartie de ce chaos est le sentiment de non-maîtrise des choses, d'évolution autonome, ce qui finalement est beaucoup plus représentatif de la réalité, personne ne pouvant se targuer de maîtriser l'ensemble d'une distribution informatique. Par ailleurs, notons que cela complique la tâche du chercheur en mettant à disposition une quantité d'informations largement supérieure à ce qu'un individu est capable de traiter. C'est aussi ce qui rend le travail d'analyse plus important et pertinent.

La redondance est fortement présente dans la communauté Ubuntu. Redondance des moyens de communication, (blogs, forums, IRC, *mailing-lists*, wiki, web, etc.), redondance des informations présentes à plusieurs endroits, comme c'est le cas pour la documentation présente sur le wiki, et qui a été transformée en manuel, mais dont les informations existent aussi dans plusieurs blogs, chaque blogueur de la communauté présentant ces informations généralement sous une forme qui lui est propre. Ici encore, il s'agit de multiplier les frontières entre disciplines, équipes, personnes différentes, enrichissant de multiples manières – sans l'a priori résultant des trajectoires technologiques (découlant elles-mêmes de choix techniques différents) – d'autres équipes.

Enfin, la variété est assurée au sein de la communauté par la diversité des origines géographique, mais aussi des champs disciplinaires des différents contributeurs à la communauté. Cette variété de personnes ayant des conceptions différentes, des connaissances différentes, est à même de fournir un vaste ensemble d'idées et de concepts pouvant donner lieu à de très nombreuses combinaisons suivant une logique factorielle (n idées donnant $n!$ possibilités de combinaisons).

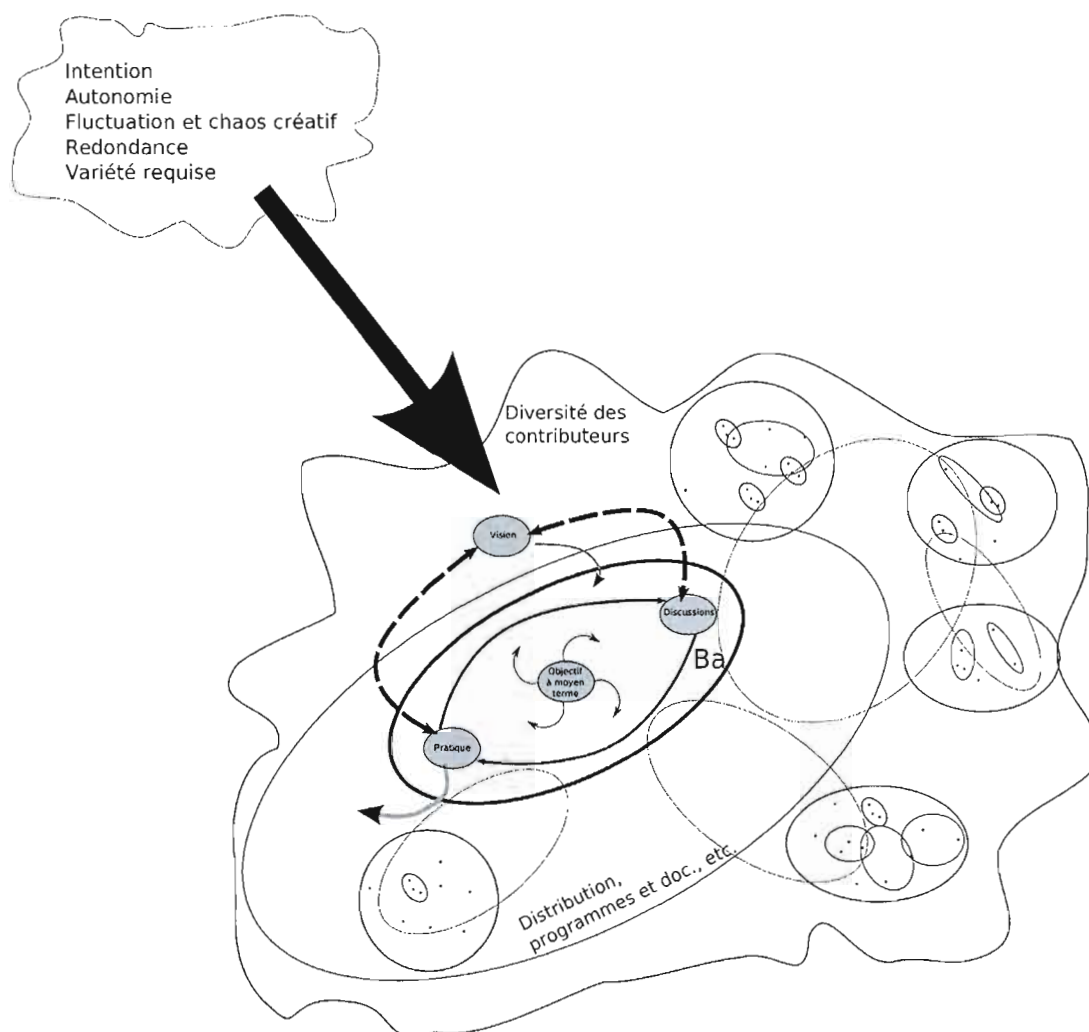


Figure 29 - Conditions de Nonaka vérifiées empiriquement dans la communauté Ubuntu

7.3 Le modèle SECI et les activateurs de connaissances

Toutes les étapes nécessaires au modèle SECI (section 1.3.2.2) ressortent assez clairement des résultats obtenus sur le terrain. Les différents forums, système de messagerie, etc. sont généralement considérés comme étant un lieu pauvre en termes de capacité de socialisation de par la « pauvreté » du médium (pas de son, pas de vidéo). Mais dans le cas du logiciel libre, notre recherche n'est pas conforme à cette description. Nous avons perçu ces systèmes comme permettant la socialisation, et

même un espace assez fort de socialisation résultant de la liberté de parole (section 6.2.1). Elle est fortement présente comme le laisse transparaître la détente des participants sur les chats de discussion ou encore sur les forums. L'autodérision semble être un activateur de cette socialisation, de même que les trolls qui en rendant polémique le sujet de discussion peuvent être à même d'être des catalyseurs de discussions. Le troll par son « décalage » par rapport à la discussion peut amener des éléments nouveaux, source de socialisation au sein des membres.

L'externalisation quant à elle est bien présente à travers la communauté, et la volonté de bien rendre le maximum de contenus explicites est forte comme le montre la citation rapportant la volonté de création d'un patron concernant l'information du fonctionnement des multiples équipes (section 6.2.3), ou la compilation de la documentation, provenant des pages du wiki (section 6.3). Les rapports de bugs, ainsi que les fiches descriptives – nécessitant argumentation et présentation de cas – servent d'artefacts efficaces pour transformer la connaissance tacite en connaissance explicite. Le code source du programme, ainsi que ses commentaires (section 6.2.2) forment aussi une source d'externalisation du concept. La création de rapport de bugs est facilitée par l'emploi de Malone rendant l'interface plus simple pour l'utilisateur, et élargissant alors la base d'utilisateurs. Les fiches descriptives ou la documentation sont quant à elles rendues plus simples par l'utilisation de wiki permettant facilement de structurer l'information.

La combinaison est assurée en partie par la diversité des membres de la communauté, appelés à travailler en commun, et dans des groupes pouvant être différents. Ceci entraîne une redondance de l'information disponible à travers les multiples blogs agrégés reprenant l'information provenant de *mailing-lists*, de forums ou de wikis. De multiples lieux de débats trouvent leurs places sur internet, et sont permis par la multiplicité des médiums (forums, wiki, launchpad, blogs, etc.), ce qui a tendance parfois à rendre l'information difficilement retraçable lorsqu'il n'est pas fait mention

de la source. Toutefois, cela est rare dans la communauté, qui généralement est sensibilisée à la question des droits d'auteurs, et donne des références. Le fait que les équipes de travail soient autogérées donne lieu aussi à de la redondance, comme nous l'avons vu section 6.2.3.

L'intériorisation, permettant la transformation de connaissance explicite en connaissance tacite, est quant à elle rendue possible par l'écriture de code ou de scripts, le fait de « *mettre les mains dans le cambouis* »¹⁶⁶ (section 6.2.2), que les utilisateurs puissent développer eux-mêmes le système qu'ils utilisent rend en quelque sorte la connaissance vivante. Par ailleurs, comme le constate Hemetsberger et Reinhardt, la présence de l'archivage des discussions portant sur les choix technique, la relecture de l'émergence de l'idée à travers la discussion entre les intervenants permet à des personnes extérieures de s'intégrer au projet en cours tout en intériorisant le travail des autres, comme dans le cas de cet intervenant <P> du projet Tribu : « *Bon ça y est, je me suis tapé les 9 pages... un début de phase d'analyse assez intéressant* »¹⁶⁷.

À travers tous ces éléments, nous décrivons un modèle de fonctionnement basé sur le modèle SECI de création de connaissance dans le cadre de la communauté virtuelle étudiée, et sommes capables de fournir des activateurs propres à la communauté Ubuntu sur la Figure 30.

¹⁶⁶ Notons qu'une recherche d'occurrence sur google de cette expression donne 42000 pages référencées, et si nous ajoutons le mot linux, nous avons 16000 pages, soit plus d'un tiers des pages.

¹⁶⁷ Source : 9^{ème} page de la discussion sur TribU sur le forum Ubuntu-fr.

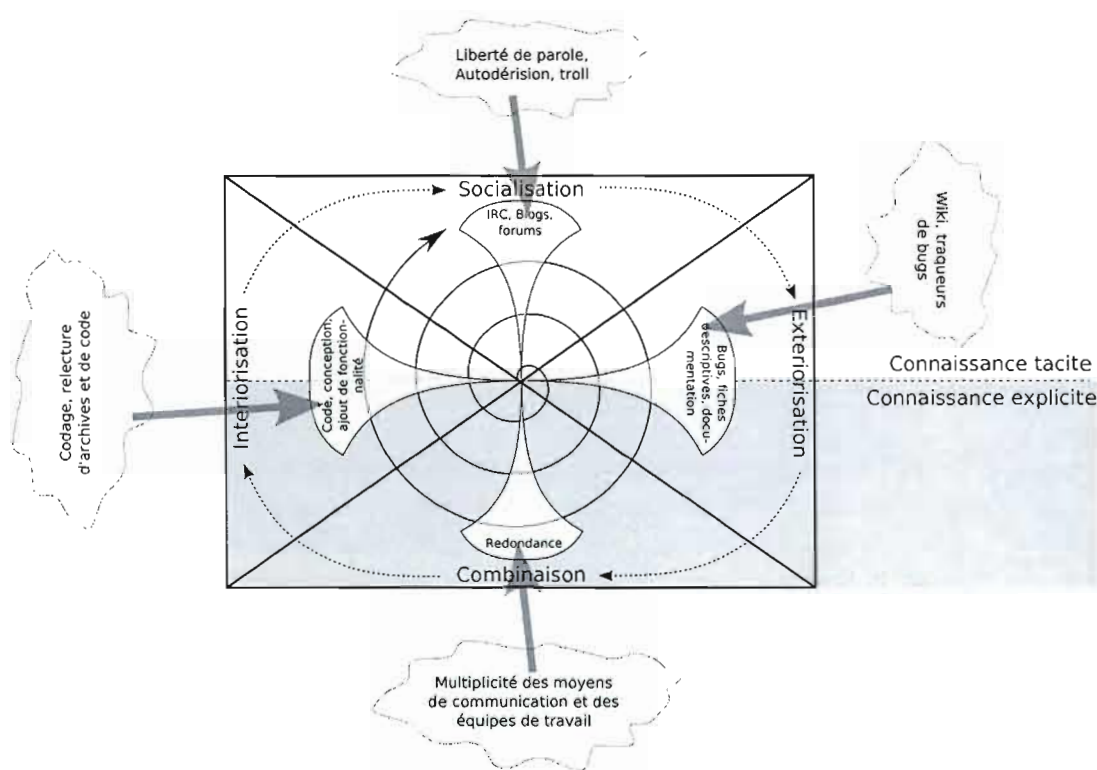


Figure 30 - Modèle SECI appliqué à Ubuntu avec activateur de connaissances

7.4 Les limites de la recherche

Malgré tous les efforts fournis dans l'élaboration de cette recherche, celle-ci souffre de deux lacunes, ayant trait à la méthode même de la recherche.

Tout d'abord, bien qu'étant sorti à de nombreuses reprises de la communauté Ubuntu, l'étude s'est centrée sur cette communauté. Il n'est donc pas possible de généraliser les résultats de la recherche à l'ensemble des communautés du logiciel libre, même si cela ne veut pas dire que nos conclusions ne soient pas valables à l'extérieur de la communauté. Il serait nécessaire, pour généraliser les résultats obtenus, de mener des études dans d'autres communautés afin de les valider.

Par ailleurs, les résultats résultent d'une enquête sur la communauté Ubuntu, une jeune communauté ayant débuté en 2004. Or, eu égard à la vitesse d'évolution de

l'écosystème du logiciel libre et de l'informatique en général, nous devons borner nos résultats à une échelle de temps raisonnable autour de la recherche.

7.5 Des perspectives d'avenir

Les résultats que nous avons été à même de faire émerger de la communauté sont extrêmement riches à notre sens. Il nous semble qu'il existe des « dimensions oubliées », qui mériteraient une nouvelle analyse, au vu d'un cadre conceptuel plus étendu, permettant de mettre en lumière ces dimensions.

Ainsi, avec la notion de cycle et de métacycle, de retards, de prolongation, de reports, etc., il nous est apparu que la dimension temps joue un rôle important dans la création de connaissance dans la communauté, favorisant peut-être plus les innovations graduelles (qui sur le long terme peuvent être spectaculaires), mais semble être aussi une faiblesse en termes d'innovation radicale nécessitant une longue phase de conception avant le démarrage du projet.

CONCLUSION

Dans ce mémoire, nous avons tenté de comprendre comment se crée la connaissance dans la communauté Ubuntu, une communauté du logiciel libre. À l'issue de notre recherche, nous avons fait apparaître une grande diversité des contributeurs, qui restent liés sous une forme communautaire par les valeurs qu'ils partagent (humanisme, bidouille, liberté, partage, autodérision, démocratisation), unis à travers la vision commune de ce qu'est l'informatique, globalement uniforme au sein de la communauté.

L'organisation interne s'effectue sur un modèle mérito-démocratique, où chacun a droit à la parole, mais où ce sont les experts et les membres les plus engagés qui ont le plus grand rôle décisionnel. La communauté est composée de nombreuses équipes autogérées, disposant d'une grande autonomie et travaillant en interaction avec d'autres équipes de la communauté Ubuntu ou d'autres communautés du logiciel libre.

Nous avons aussi eu l'occasion de montrer l'importance des cycles et des métacycles de création de connaissances, permettant de choisir une trajectoire technologique puis de s'en écarter par la suite.

À partir de ces résultats, nous avons proposé un modèle de création de connaissance, basé sur les modèles de Nonaka et Takeuchi (1995) et Nonaka et Toyama (2005). Nous avons aussi été amenés à valider les activateurs de connaissances de Nonaka (intention, variété, chaos créatif, redondance, autonomie), et à déterminer d'autres activateurs de création de connaissances propres au logiciel libre, à savoir la liberté de parole, la possibilité de *troller*, les outils d'extériorisation (du type wiki, bugzilla,

etc.), la diversité des moyens de communication et des contributeurs et l'action même de coder.

Pour toutes ces raisons, l'univers du logiciel libre a le formidable avantage de posséder une dynamique importante, d'être un lieu (virtuel) de perpétuel mouvement, de création continue, de tissage de liens entre individus.

Il nous semble que le principal danger qu'encourt le logiciel libre est la volonté affichée par certains – mais jusque-là écartée – de la mise à l'écart de ses idéaux au profit d'une plus grande efficacité *de fait*, de privilégier l'idée de construire des logiciels libres non plus pour promouvoir la liberté informatique, mais pour l'efficacité même du modèle en termes de création de connaissances.

Si le changement peut sembler gagnant à court terme, à long terme, nous pensons que le libre en général y serait perdant, car il n'y aurait plus d'esprit du libre, donc plus de distinction par rapport au monde propriétaire, assurant à notre avis, un lent déclin de cet écosystème aujourd'hui si dynamique. Il ne s'agit pas d'un futur probable, mais il s'agit d'un futur possible.

Pour l'intérêt du logiciel libre en général, de ses utilisateurs et de ses développeurs, souhaitons qu'un jour, il ne s'agisse plus que d'un futur antérieur à jamais révolu.

BIBLIOGRAPHIE GÉNÉRALE

- Argyris, Chris, et Donald A. Schön. 1978. *Organizational learning*, 2 t. Reading, Mass.: Addison-Wesley, tome I, 344 p.
- Artus, Patrick. 2000. «Nouvelle économie, nouveaux problèmes». In *Espérances et menaces de la nouvelle économie*, Le cercle des économistes, p. 478. Paris: Descartes & Cie.
- Auray, Nicolas, et Mickaël Vicente. 2006. *Engagement dans le logiciel libre comme réaction au moment intégré, Etude longitudinale de l'engagement dans les projets Debian et OpenBSD*. (74e congrès de l'ACFAS, Université McGill, Montréal - Le logiciel libre en tant que modèle d'innovation sociotechnique : pratiques de développement et de coopération dans les communautés).
- Ballay, Jean-François. 2002. *Tous managers du savoir! : la seule ressource qui prend de la valeur en la partageant*. Paris: Éditions d'Organisation, 430 p.
- Barbrook, Richard. 2000. «L'économie du don High-Tech». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Berlan, Jean-Pierre. 2003. *A propos de la propriété intellectuelle* (Parlement Européen, 28 avril 2003).
- Bessen, James, et Eric Maskin. 1999. «Sequential innovation, patents, and imitation». *Working Paper, Department of Economics, Massachusetts Institute of Technology*.
- Blondeau, Olivier. 2000a. «Genèse et subversion du capitalisme informationnel». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Blondeau, Olivier. 2000b. «Les États du Libre». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.

- Blondeau, Olivier. 2000c. «Une nouvelle Économie... du savoir?». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Bourhis, Anne , Line Dubé et Réal Jacob. 2004. «La contribution de la gestion des connaissances à la gestion de la relève : le cas Hydro-Québec». *Cahier du GReSI*, no 04-12, p. 18 pages.
- Bourhis, Anne, et Diane-Gabrielle Tremblay. 2004. *Les facteurs organisationnels de succès des communautés de pratique virtuelles : projet Modes de travail et de collaboration à l'ère d'Internet : rapport de recherche présenté au CEFRIO*. Québec: Centre francophone d'informatisation des organisations, CEFRIO, 173 p.
- Breton, Philippe, et Serge Proulx. 1989. *L'explosion de la communication la naissance d'une nouvelle ideologie*. Coll. «Sciences et société». Paris: La Découverte, 285 p.
- Caillaud, Bernard. 2003. «La propriété intellectuelle sur les logiciels». In *Propriété intellectuelle*, Jean Tirolle. Paris: La documentation française.
- Centre pour la recherche et l'innovation dans l'enseignement, et OCDE. 2000. *Société du savoir et gestion des connaissances*. Paris: Organisation de coopération et de développement économiques, 293 p. En ligne. <<http://accesbib.uqam.ca/cgi-bin/bduqam/transit.pl?&noMan=18037613>>.
- Chabchoub, Norhène. 2005. *L'industrie nord-américaine du logiciel : protection intellectuelle et diffusion de la pratique des brevets*. Montréal: Université du Québec à Montréal, 242 p. En ligne. <<http://accesbib.uqam.ca/cgi-bin/bduqam/transit.pl?&noMan=24144715>>.
- Chatelain, Yannick. 2006. «Management of Innovation, the Internet and Deviance: a typology for the integration of *hacker* logic by companies.». University of Newcastle, Grenoble Ecole de Management.
- Chollet, Barthélémy. 2004. «Le rôle du capital social de l'ingénieur R&D, une étude exploratoire dans le secteur de la microélectronique». *AIMS*, p. 26.
- Cohen, Daniel. 2003. «Commentaire». In *Propriété intellectuelle*, Jean Tirole, Claude Henry, Michel Trommetter, Laurence Tubiana, Bernard Caillaud, Daniel Cohen et Lionel Fontagné, p. 173-175. Paris: La documentation française, Conseil d'analyse économique.

- Cohendet, Patrick, Frédéric Creplet et Olivier Dupouët. 2003. «Innovation organisationnelle, communautés de pratique et communautés épistémiques : le cas de Linux». *Revue Française de Gestion*, no 146, p. 98-124.
- Couture, Stéphane. 2006. *La construction des modèles du libre* (74e congrès de l'ACFAS, Université McGill, Montréal - Le logiciel libre en tant que modèle d'innovation sociotechnique : pratiques de développement et de coopération dans les communautés).
- Crowston, Kevin , James Howison et Hala Annabi. 2006. «Information Systems Success in Free and Open Source Software Development: Theory and Measures». *Software process improvement and practice*, no 11, p. 123-148.
- Crowston, Kevin, et James Howison. 2005. «The social structure of Free and Open Source software development». *First Monday*, February.
- Davenport, Thomas H., et Donald A. Marchand. 2000. «De l'information à la connaissance». In *L'art du management de l'information*, Financial Times Business Information Ltd. et Les Échos (Journal), p. ix, 373. Paris: Village mondial.
- Demazière, Didier, François Horn et Marc Zune. 2006. *La contribution aux logiciels libres comme engagement : une approche polycentrique des communautés du logiciel libre* (74e congrès de l'ACFAS, Université McGill, Montréal - Le logiciel libre en tant que modèle d'innovation sociotechnique : pratiques de développement et de coopération dans les communautés).
- Didier, Michel. 2000. «Nouvelles économies, d'hier et d'aujourd'hui». In *Espérances et menaces de la nouvelle économie*, Le cercle des économistes, p. 478. Paris: Descartes & Cie.
- Dubé, Line, Anne Bourhis et Réal Jacob. 2003. «Towards a Typology of Virtual Communities of Practice». *Cahier du GReSI No 03-27*, p. 27.
- Ebrahimi, Mehran, et Anne-Laure Saives. 2006. «Le management de l'innovation et des connaissances: de l'ère industrielle à celle du savoir et de l'intangible». In *Le management entre tradition et renouvellement*, 4e éd., Omar Aktouf, p. 663. Montréal: Gaëtan Morin.

- Ferrary, Michel, et Yvon Pesqueux. 2004. *L'organisation en réseau : mythes et réalités*, 1re éd. Coll. «La politique éclatée». Paris: Presses universitaires de France, 293 p.
- Foray, Dominique. 2000. *L'économie de la connaissance*. Coll. «Collection Repères 302». Paris: La Découverte, 123 p.
- Foray, Dominique, et Jean-Benoît Zimmermann. 2001. «L'économie du logiciel libre : organisation coopérative et incitation à l'innovation». *Revue Economique*. vol. 51, p. 77-93.
- Forsyth, Donelson R. 1990. *Group Dynamics*. USA: Pacific Grove.
- FSF. 2004. «Qu'est ce qu'un logiciel libre». En ligne. <<http://www.gnu.org/philosophy/free-sw.fr.html>>.
- Gadrey, Jean. 2000. *Nouvelle économie, nouveau mythe? ; suivi de Que reste-t-il de la nouvelle économie?* Coll. «Champs (Flammarion (Firme)) 489». Paris: Flammarion, 267 p.
- Gibbons, Michael. 1994. *The New production of knowledge the dynamics of science and research in contemporary societies*. London: Sage, ix, 179 p.
- Goldenberg, Anne. 2006. *Extension(s) du domaine du libre* (74e congrès de l'ACFAS, Université McGill, Montréal - Le logiciel libre en tant que modèle d'innovation sociotechnique : pratiques de développement et de coopération dans les communautés).
- Gusdorf, Georges. 1974. *Introduction aux sciences humaines, préface pour l'édition italienne*. Paris: Éditions Ophrys.
- Heilbroner, Robert Louis. 1977. *Les grands économistes*. Coll. «Points. Économie E7». Paris: Éditions du Seuil, 335 pages p.
- Hemetsberger, Andrea, et Christian Reinhardt. 2004. «Sharing and Creating Knowledge in Open-Source Communities. The case of KDE». *The Fifth European Conference on Organizational Knowledge, Learning, and Capabilities, Innsbruck, Austria*. En ligne. <<http://opensource.mit.edu/papers/hemreinh.pdf>>.

- Hemetsberger, Andrea, et Christian Reinhardt. 2006 (à paraître). «Learning and Knowledge-building in Open-source Communities». *Management Learning*, vol. 37, no 2, p. 187-214.
- Henry, Claude, Michel Trommetter et Laurence Tubiana. 2003. «Innovations et droits de propriété intellectuelle : quels enjeux pour les biotechnologies ?». In *Propriété intellectuelle, Conseil d'analyse économique*, Jean Tirole, Claude Henry, Michel Trommetter, Laurence Tubiana, Bernard Caillaud, Daniel Cohen et Lionel Fontagné, p. 49-112. Paris: La documentation française.
- Himanen, Pekka. 2001. *L'éthique hacker et l'esprit de l'ère de l'information*. Coll. «Essais (Exils éditeur)». Paris: Exils, 219 p.
- Horn, François. 2000. «L'économie du logiciel». thèse de doctorat d'Économie industrielle, Lille, Faculté des Sciences Économiques et Sociales, Université des Sciences et Technologies de Lille, 570 p.
- Jacquet, Pierre. 2000. «Nouvelle économie et gouvernance de la mondialisation». In *Espérances et menaces de la nouvelle économie*, Le cercle des économistes, p. 478. Paris: Descartes & Cie.
- Jullien, Nicolas, et Jean-Benoît Zimmermann. 2005. «Peut-on envisager une écologie du Libre favorable aux nuls?». *GREQAM, Document de Travail n°2005-51*.
- Kline, S.J., et Nathan Rosenberg. 1986. «An overview of innovation». In *The Positive sum strategy : harnessing technology for economic growth*, Ralph Landau, Nathan Rosenberg et National Academy of Engineering., p. xiv, 640. Washington, D.C.: National Academy Press.
- Lainesse, Vicky. 2003. «La création d'un noeud de communication : le cas d'un réseau d'infirmières». Montréal, Canada, Département de communication - Faculté des arts et Sciences de l'Université de Montréal.
- Lang, Bernard. 2000. «Le nouveau protectionnisme est intellectuel». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Laperrière, Anne. 1997. «Les critères de scientificité des méthodes qualitatives». In *La recherche qualitative enjeux épistémologiques et méthodologiques*, Jean Poupart et Groupe de recherche interdisciplinaire sur les méthodes qualitatives, p. 405. Montréal: G. Morin.

- Latrive, Florent. 2004. *Du bon usage de la piraterie : culture libre, sciences ouvertes*. Coll. «Essais». Paris: Exils Editeur, 170 p.
- Lave, J , et Etienne Wenger. 1991. *Situated Learning : legitimate peripheral participation*. Combridge, UK: Cambridge University Press.
- Lee, Gwendolyn K., et Robert E. Cole. 2003. «From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development». *Organization Science*. vol. 14, no 6, p. 633.
- Levy, Ron. 1994. «Croyance et doute: une vision paradigmatique des méthodes qualitatives». *Ruptures, revue transdisciplinaire en santé*. vol. 1, no 1, p. 92-100.
- Liotard, Isabelle. 1999. «Les droits de propriété intellectuelle – une nouvelle arme stratégique des firmes». *Revue d'économie industrielle*, no 89, p. 69-84.
- Lundvall, Bengt-Åke. 2000. «L'économie apprenante et certaines de ses conséquences pour la base de savoir du système de santé et du système éducatif». In *Société du savoir et gestion des connaissances*, p. 293. Paris: Centre pour la recherche et l'innovation dans l'enseignement, OCDE. En ligne. <<http://accesbib.uqam.ca/cgi-bin/bduqam/transit.pl?&noMan=18037613>>.
- Mandel, Michael J. . 1996. «The triumph of the new economy ». *Business Week* (New York), 30 december. En ligne. <<http://www.businessweek.com/1996/53/b35081.htm>>.
- Mauss, Marcel. 1923-1924. *Essai sur le don. Forme et raison de l'échange dans les sociétés archaïques*. Chicoutimi: Les classiques des sciences sociales. En ligne. <http://www.uqac.quebec.ca/zone30/Classiques_des_sciences_sociales/index.html>.
- McDermott, R , et R J Jackson. 2002. «Global Knowledge». *Cutter IT journal*.
- Molinié, Luis. 2006. *La portabilité du libre : le cas Wikipédia* (74e congrès de l'ACFAS, Université McGill, Montréal - Le logiciel libre en tant que modèle d'innovation sociotechnique : pratiques de développement et de coopération dans les communautés).

- Nonaka, Ikujiro, et Noboru Konno. 1998. «The concept of "Ba"». *California Management Review*. vol. 3, no 40, p. 40.
- Nonaka, Ikujiro, et Hitoshi Takeuchi. 1997. *La connaissance créatrice la dynamique de l'entreprise apprenante*. Coll. «Management». Paris: De Boeck Université, 303 p.
- Nonaka, Ikujiro, et Ryoko Toyama. 2003. «The knowledge-creating theory revisited: knowledge creation as a synthesizing process». *Knowledge Management Research & Practice*. vol. 1, p. 2-10.
- Nonaka, Ikujiro, et Ryoko Toyama. 2005. «The theory of knowledge-creating firm: subjectivity, objectivity and synthesis». *Industrial and Corporate Change*. vol. 14, no 3, p. 419-436.
- OSI. 1997-2006. «The Open Source Definition». En ligne. <<http://www.opensource.org/docs/definition.php>>.
- Paillé, Pierre. 1994. «L'analyse par théorisation ancrée». *Cahiers de recherche sociologique*, no 23, p. 147-181.
- Pesqueux, Yvon. 2002. *Organisations : modèles et représentations*. Paris: Presses universitaires de France, 396 p.
- Plihon, Dominique. 2003. *Le nouveau capitalisme*. Paris: La Découverte, 121 p.
- Polanyi, Michael. 1966. *The tacit dimension*. Coll. «Terry lectures 1962». Garden City, N.Y.: Doubleday, 108 p.
- Prax, Jean-Yves. 2000. *Le guide du knowledge management concepts et pratiques du management de la connaissance*. Coll. «Stratégies et management». Paris: Dunod, 266 p.
- Quéau, Philippe. 2000. «Intérêt général et propriété intellectuelle». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Raymond, Eric S. 1998. «La cathédrale et le bazar». *Traducteur : Sébastien Blondeel*. En ligne. <<http://www.linux-france.org/article/these/cathedrale-bazar/cathedrale-bazar.html>>.

- Raymond, Eric S. 1999. «À la conquête de la noosphère». *Traducteurs : Emmanuel Fleury et Sébastien Blondeel*. En ligne. <<http://www.linux-france.org/article/these/noosphere/>>.
- Raymond, Eric S. 2000. «Comment devenir un *hacker*». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Samudrala, Ram. 2000. «Créativité et propriété, où est le juste milieu». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504: Edition de l'éclat.
- Senge, Peter M. 1991. *La cinquième discipline l'art et la manière des organisations qui apprennent*. Paris: First, 462 p.
- Stallman, Richard M. 1999. «Le système d'exploitation du projet GNU et le mouvement du logiciel libre». *Traducteur : Sébastien Blondeel*. En ligne. <<http://www.linux-france.org/article/these/gnuproject/fr-thegnuproject.html>>.
- Stallman, Richard M. 2000. «Pourquoi les logiciels ne doivent pas avoir de propriétaire». En ligne. <<http://www.gnu.org/philosophy/why-free.fr.html>>.
- Statistique Canada. 2004. «Enquête sur l'utilisation d'Internet par les ménages». *Le Quotidien*. En ligne. <<http://www.statcan.ca/Daily/Francais/040708/q040708a.htm>>.
- Statistique Canada. 2005. «Enquête sur les dépenses des ménages». *Le Quotidien*. En ligne. <<http://www.statcan.ca/Daily/Francais/051212/q051212a.htm>>.
- Sterling, Bruce. 2000. «Libre comme l'eau, l'air, le savoir». In *Libres enfants du savoir numérique*, Olivier Blondeau et Florent Latrive, p. 504. Perreux, France: Edition de l'éclat.
- Stewart, Katherine J., et Sanjay Gosain. 2006 (à paraître). «The impact of ideology on effectiveness in open source software development teams». *MIS Quarterly*.
- Stoffaës, Christian. 2000. «Nouveaux réseaux et nouveaux services publics». In *Espérances et menaces de la nouvelle économie*, Le cercle des économistes, p. 478. Paris: Descartes & Cie.

- Tisseyre, René-Charles. 1999. *Knowledge management théorie et pratique de la gestion des connaissances*. Coll. «Collection Nouvelles technologies informatiques». Paris: Hermès, 185 p.
- Torvalds, Linux. 2001. «Préface». In *L'éthique hacker et l'esprit de l'ère de l'information*, Pekka Himanen, p. 219. Paris: Exils.
- Twidale, Michael B., et David M. Nichols. 2005. «Exploring usability discussions in open source software». *Proceedings of the Thirty-Eighth Annual Hawaii International Conference on System Sciences (HICSS'05)*. En ligne. <<http://csdl.computer.org/comp/proceedings/hicss/2005/2268/07/22680198c.pdf>>.
- Von Hippel, Eric. 2005. *Democratizing innovation*. Cambridge.
- Von Krogh, Georg, Ichijo Kazuo et Ikujiro Nonaka. 2000. *Enabling knowledge creation*. New York: Oxford University Press.
- Weber, Steve. 2004. *The success of open source*. Cambridge, Mass. ; London: Harvard University Press, viii, 312 p.
- Wenger, Etienne. 1998a. *Communities of Practice : Learning, Meaning, and Identity*. New York, USA: Cambridge University Press.
- Wenger, Etienne , R McDermott et William Snyder. 2002. *Cultivating Communities of Practice*. Boston, USA: Harvard Business School Press.
- Wenger, Etienne, et William Snyder. 2003. «Des communautés de pratique - Le nouvel horizon organisationnel». In *Les meilleurs articles de la Harvard Business Review sur le Management du savoir en pratique*, p. 91-115. Paris, France: Editions d'Organisation.
- Wheeler, David A. . 2005. «Why Open Source Software / Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!». http://www.dwheeler.com/oss_fs_why.html.
- Wikipedia. a. «Paradigme». En ligne. <<http://fr.wikipedia.org/wiki/Paradigme>>.

ANNEXE 1

LISTE DES *MAILING-LISTS* UBUNTU

“Welcome!

Below is a listing of all the public mailing lists on lists.ubuntu.com. Click on a list name to get more information about the list, or to subscribe, unsubscribe, and change the preferences on your subscription. To visit the general information page for an unadvertised list, open a URL similar to this one, but with a '/' and the list name appended.

List administrators, you can visit the list admin overview page to find the management interface for your list.

If you are having trouble using the lists, please contact mailman@lists.ubuntu.com.

<u>List</u>	<u>Description</u>
bazaar	User and development discussion about the bazaar distributed revision control system
bazaar-announce	Announcements for the Bazaar project
bazaar-commits	Bazaar repository commit notification
bazaar-ng	bazaar-ng discussion
breezy-changes	BreezyBadger archive upload notification list
dapper-changes	DapperDrake archive upload notification list
desktop-bugs	Desktop bug tracker changes - HIGH VOLUME
edgy-changes	EdgyEft archive upload notification list

edubuntu-devel	Edubuntu development discussion
edubuntu-devel-es	Edubuntu Hispanoparlantes
edubuntu-users	Edubuntu Users Help and Discussion
hct	Discussion about the Hypothetical Changeset Tool
hoary-changes	HoaryHedgehog archive upload notification list
kernel-bugs	Kernel Bugs tracking
kernel-team	Kernel team discussions
kubuntu-bugs	Kubuntu bug tracker changes
kubuntu-devel	Kubuntu Developer Discussion
kubuntu-users	Kubuntu Help and User Discussions
laptop-devel	laptop-specific development
laptop-testing-team	Ubuntu laptop testing
launchpad-users	Discussion for Launchpad users
loco-contacts	Ubuntu local community team (LoCo) contacts
rosetta-users	Rosetta user discussion
security-review	Discussion about resolving security vulnerabilities
sounder	Ubuntu community random chit-chat list
Ubuntu-accessibility	mailing list of the accessibility team
ubuntu-announce	Ubuntu Announcements
ubuntu-archive	Ubuntu Archive Team discussion and co-ordination
ubuntu-art	Discussion on Ubuntu artwork
ubuntu-au	Ubuntu in Australia
ubuntu-backports	Backports Discussion
ubuntu-bd	Ubuntu Bangladesh
ubuntu-be	Ubuntu Belgium
ubuntu-br	Lista de discussão do LoCoTeam Brasileiro
ubuntu-bugs	Ubuntu bug tracker changes - HIGH VOLUME
ubuntu-ca	The Canadian Ubuntu Users Community

ubuntu-changes-auto	Archive upload notification list, for automated uploads to all Ubuntu releases
ubuntu-de	Ubuntu-Diskussion
ubuntu-desktop	Desktop Team co-ordination and discussion
ubuntu-devel	Ubuntu Developer Discussion
ubuntu-devel-announce	Developer-related announcements and information
ubuntu-dk	Dansk mailingliste til det danske Ubuntu LocoTeam til diskussion/koordination af aktiviteter
ubuntu-doc	Documentation team coordination and discussion
ubuntu-doc-commits	Ubuntu documentation team commits
ubuntu-es	Asistencia para usuarios de Ubuntu
Ubuntu-es-l10n	List for coordination of Ubuntu Spanish translation team
ubuntu-fi	Sähköpostilista Ubuntu-keskustelulle
ubuntu-fr	Liste de discussion sur Ubuntu
Ubuntu-gr	Greek Ubuntu Users Discussion List
ubuntu-hardened	Ubuntu Linux Proactive Security deployment and development
ubuntu-id	Ubuntu User Group Indonesia.
ubuntu-il	Israel's Ubuntu Linux Mailing List
ubuntu-in	Ubuntu India Local Community
Ubuntu-irc	Ubuntu IRC issues discussion list
ubuntu-it	Supporto per gli utenti Italiani di UbuntuLinux
Ubuntu-l10n-am	[no description available]
Ubuntu-l10n-gl	List for coordination of Ubuntu Galician translation team
Ubuntu-l10n-hu	List for coordination of Ubuntu Hungarian translation team
Ubuntu-l10n-oc	Revirar Ubuntu en occitan...
ubuntu-l10n-ru	List for coordination of Ubuntu Russian translation team
ubuntu-marketing	Discussion on community-based marketing of Ubuntu

ubuntu-mirrors	Announcements, feedback and discussion for Ubuntu mirror maintainers
Ubuntu-mirrors-announce	Ubuntu Mirror Annoucement List
Ubuntu-mono	Packaging Mono for Ubuntu
Ubuntu-motu	mailing list of the Masters Of The Universe
ubuntu-news	Interesting news about Ubuntu for users and developers
ubuntu-nl	Nederlandstalige Ubuntu-ondersteuning en discussie
ubuntu-no	Generell epost-liste for Ubuntu Norge
ubuntu-patches	Ubuntu patch upload notifications - HIGH VOLUME
ubuntu-ph	Mailing List para sa Ubuntu Pilipinas (Philippines)
ubuntu-pl	Polska lista dyskusyjna użytkowników Ubuntu
ubuntu-pl-l10n	List for coordination of Ubuntu Polish translation team
ubuntu-pt	Lista de Discussão sobre o Ubuntu
ubuntu-ro	Lista de discutii despre Ubuntu
ubuntu-ru	Mailing list for Russian-speaking Ubuntu users
ubuntu-se	Ubuntu Sverige
ubuntu-security-announce	Ubuntu Security Announcements
ubuntu-translators	Discussion about translating Ubuntu
ubuntu-uk	British Ubuntu Talk
ubuntu-users	Ubuntu Help and User Discussions
ubuntu-utah	ubuntu-utah mailing list
ubuntu-women	Ubuntu-Women
ubuntu-za	Ubuntu South African Local Community
ubuntu-zh	ubuntu-zh mailing lists
universe-bugs	Universe bug tracker changes - HIGH VOLUME
warty-changes	WartyWarthog archive upload notification list
xubuntu-devel	Xubuntu Development Discussion"

Source: <https://lists.ubuntu.com/mailman/listinfo/>

ANNEXE 2

CODE OF CONDUCTS UBUNTU

“= Ubuntu Code of Conduct =

This Code of Conduct covers your behaviour as a member of the Ubuntu Community, in any forum, mailing list, wiki, web site, IRC channel, install-fest, public meeting or private correspondence. The Ubuntu Community Council will arbitrate in any dispute over the conduct of a member of the community.

""Be considerate."" Your work will be used by other people, and you in turn will depend on the work of others. Any decision you take will affect users and colleagues, and we expect you to take those consequences into account when making decisions. For example, when we are in a feature freeze, please don't upload dramatically new versions of critical system software, as other people will be testing the frozen system and will not be expecting big changes.

""Be respectful."" The Ubuntu community and its members treat one another with respect. Everyone can make a valuable contribution to Ubuntu. We may not always agree, but disagreement is no excuse for poor behaviour and poor manners. We might all experience some frustration now and then, but we cannot allow that frustration to turn into a personal

attack. It's important to remember that a community where people feel uncomfortable or threatened is not a productive one. We expect members of the Ubuntu community to be respectful when dealing with other contributors as well as with people outside the Ubuntu project and with users of Ubuntu.

"Be collaborative." Ubuntu and Free Software are about collaboration and working together. Collaboration reduces redundancy of work done in the Free Software world, and improves the quality of the software produced. You should aim to collaborate with other Ubuntu maintainers, as well as with the upstream community that is interested in the work you do. Your work should be done transparently and patches from Ubuntu should be given back to the community when they are made, not just when the distribution releases. If you wish to work on new code for existing upstream projects, at least keep those projects informed of your ideas and progress. It may not be possible to get consensus from upstream or even from your colleagues about the correct implementation of an idea, so don't feel obliged to have that agreement before you begin, but at least keep the outside world informed of your work, and publish your work in a way that allows outsiders to test, discuss and contribute to your efforts.

"When you disagree," consult others. Disagreements, both political and technical, happen all the time and the Ubuntu community is no exception. The important goal is not to avoid disagreements or differing views but to resolve them constructively. You should turn to the community and to the

community process to seek advice and to resolve disagreements. We have the Technical Board and the Community Council, both of which will help to decide the right course for Ubuntu. There are also several Project Teams and Team Leaders, who may be able to help you figure out which direction will be most acceptable. If you really want to go a different way, then we encourage you to make a derivative distribution or alternative set of packages available using the Ubuntu Package Management framework, so that the community can try out your changes and ideas for itself and contribute to the discussion.

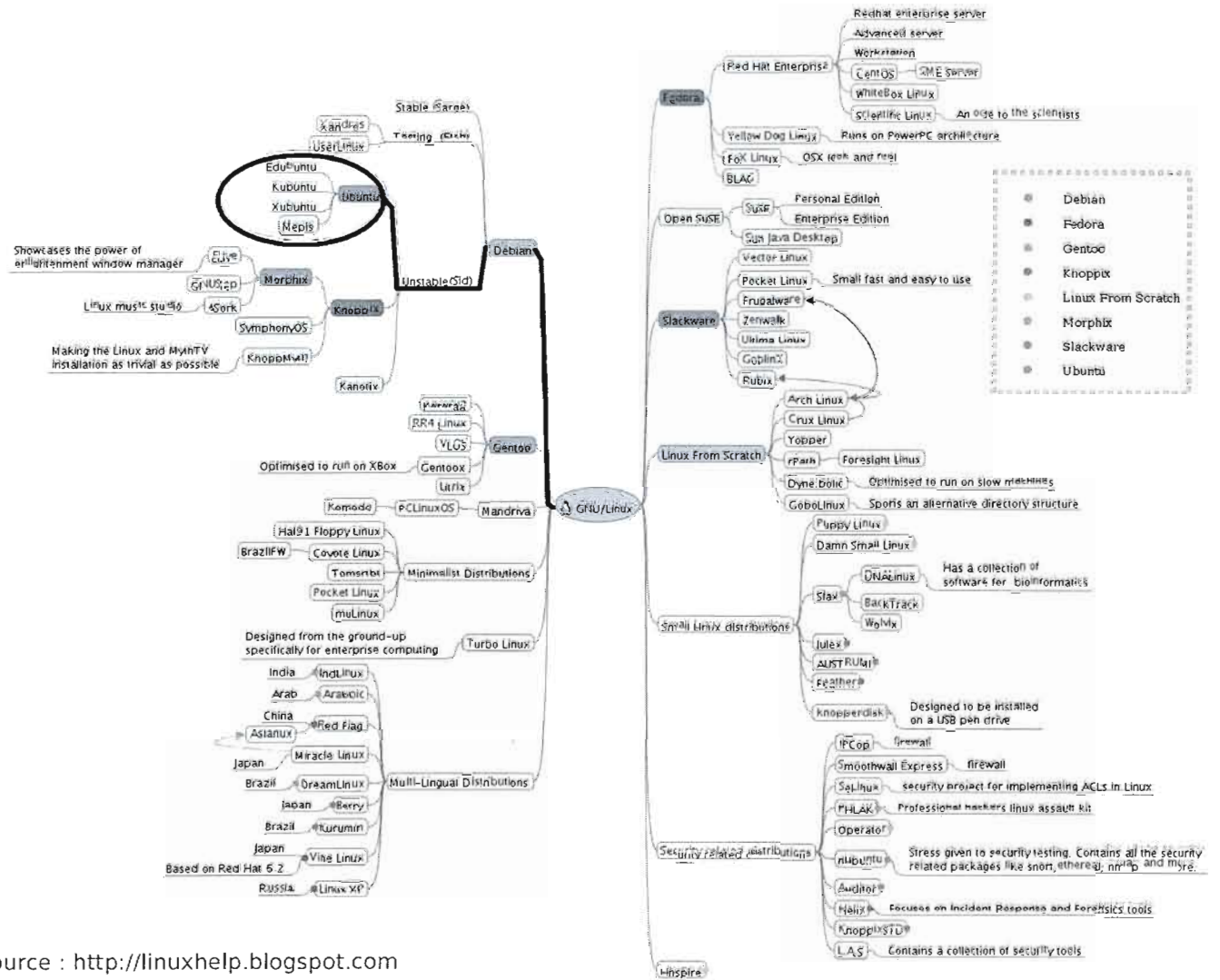
"When you are unsure," ask for help. Nobody knows everything, and nobody is expected to be perfect in the Ubuntu community (except of course the SABDFL). Asking questions avoids many problems down the road, and so questions are encouraged. Those who are asked should be responsive and helpful. However, when asking a question, care must be taken to do so in an appropriate forum. Off-topic questions, such as requests for help on a development mailing list, detract from productive discussion.

"Step down considerably." Developers on every project come and go and Ubuntu is no different. When you leave or disengage from the project, in whole or in part, we ask that you do so in a way that minimises disruption to the project. This means you should tell people you are leaving and take the proper steps to ensure that others can pick up where you leave off."¹⁶⁸

¹⁶⁸ <https://launchpad.net/codeofconduct/1.0>

ANNEXE 3

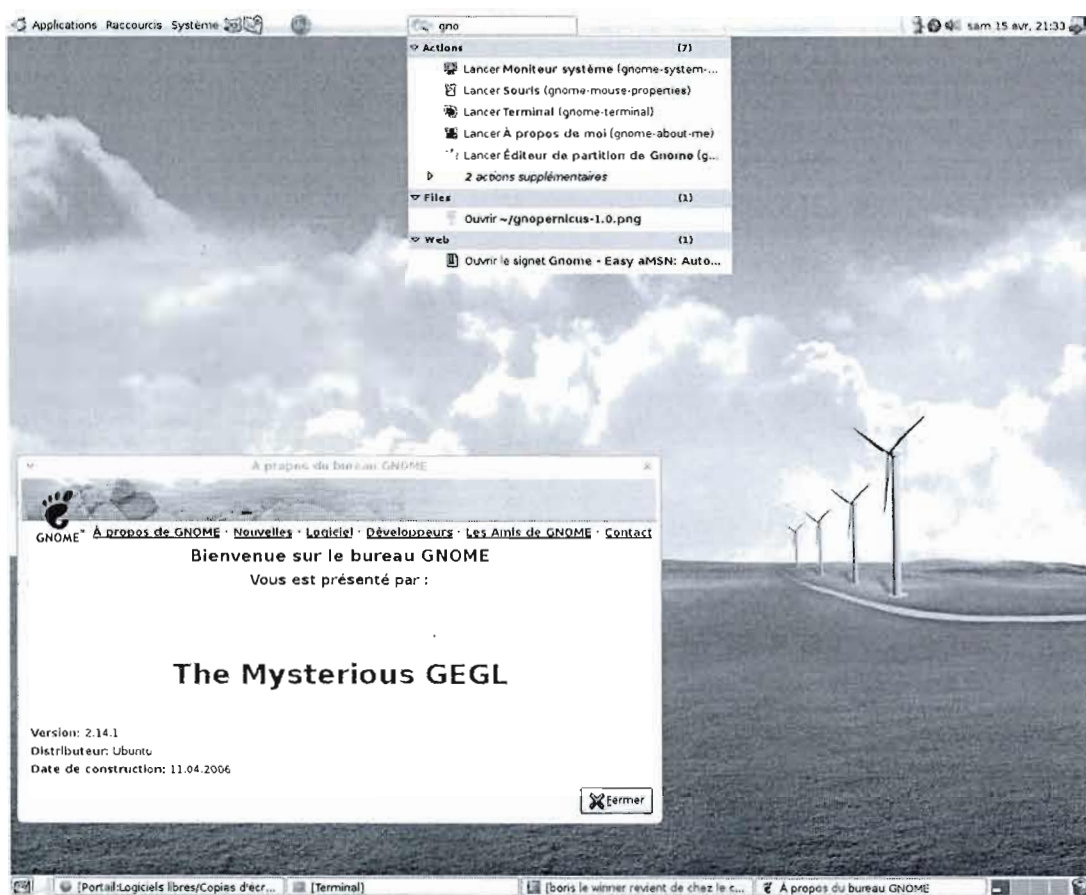
“MIND MAP” DES PRINCIPALES DISTRIBUTIONS LINUX

Source : <http://linuxhelp.blogspot.com>

ANNEXE 4

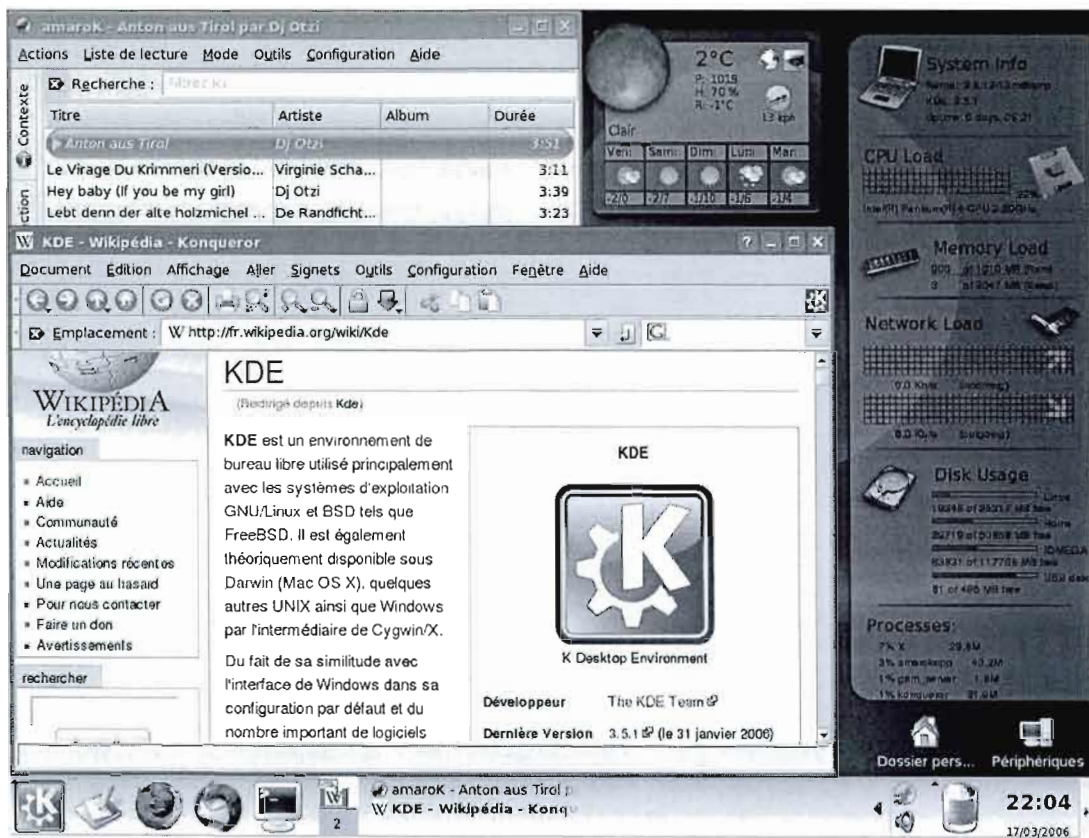
COPIES D'ÉCRAN DIVERSES

Copie d'écran d'un bureau Gnome :



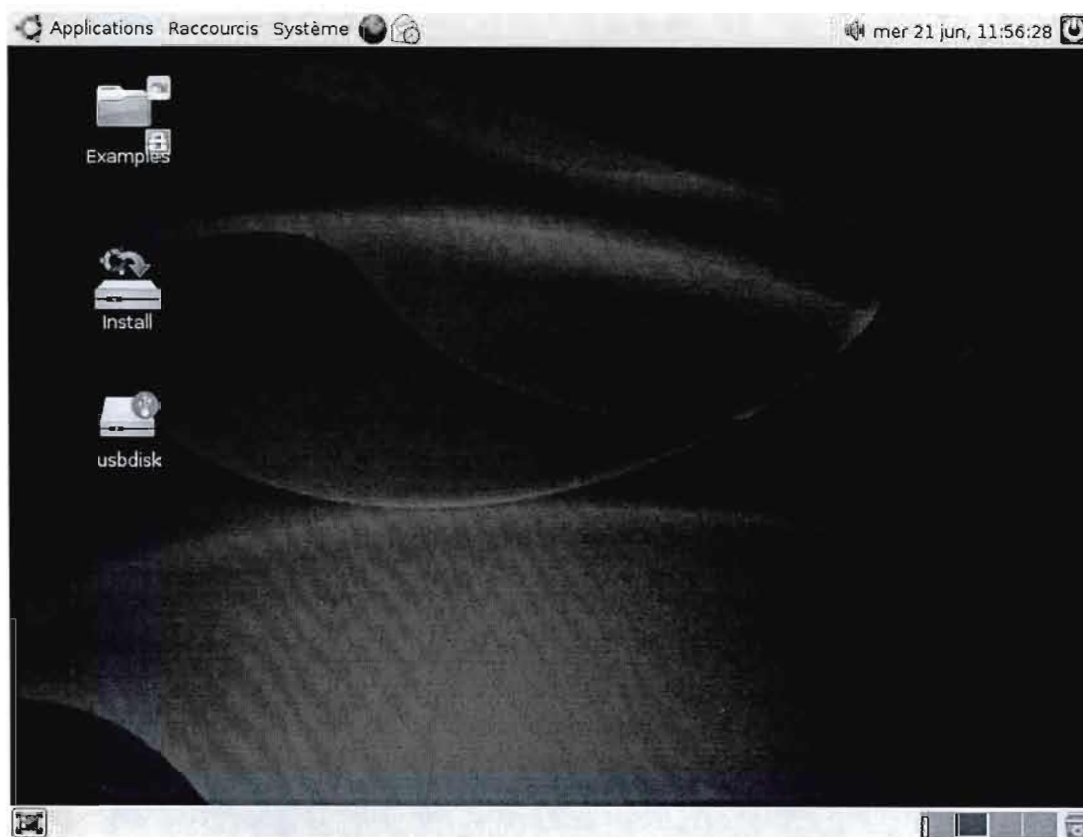
(source : wikipédia)

Copie d'écran de d'un bureau KDE :



(source : wikipédia)

Copie d'écran d'un bureau Ubuntu (version Dapper utilisant GNOME 2.14) :




(source : wikipedia)

ANNEXE 5







SPECIFICATIONS

FEATURESPECIFICATIONS¹⁶⁹

This page describes the process by which we define, discuss, schedule and implement new feature specifications in Ubuntu and Launchpad.

Ubuntu and Launchpad are both complex software projects with community involvement to various degrees. We need to keep track of many different initiatives, upstream and in the distribution, and we need to be able to know exactly where we stand on the delivery of those during the release cycle. We use the  Launchpad Specification Tracker to do this.

THE SPECIFICATION PROCESS

1. **Write your specification in the wiki.** There are currently separate wikis for Launchpad  here and Ubuntu (you are in it now :-)). Take a look at the existing specifications to ensure that you are not duplicating an existing proposal. When you create the page, you are offered a list of Page Templates; use the one called SpecTemplate. The spec should be as detailed as you can make it: best practices documented are on the SpecSpec.
2. **Register your specification in Launchpad.** Specifications should be attached to either Ubuntu or a Launchpad product. Use the following quick URLs:
 - a.  <https://launchpad.net/distros/ubuntu/+specs> (Ubuntu, Kubuntu and Edubuntu specs)
 - b.  <https://launchpad.net/products/rosetta/+specs> (Rosetta specs)
 - c.  <https://launchpad.net/products/malone/+specs> (Malone specs)
 - d.  <https://launchpad.net/products/launchpad/+specs> (General launchpad specs)
 - e.  <https://launchpad.net/products/bzr/+specs> (Bazaar-NG specs)

¹⁶⁹ Source : <https://wiki.ubuntu.com/FeatureSpecifications>

- The URLs above will list specs already registered. Use the "Register New Specification" link to add your spec. If you will be coding or implementing this yourself, make yourself the Assignee; otherwise, leave it blank.

If you are the person responsible for drafting the specification and getting it to the point where it is approved, make yourself the Drafter. If there is someone else who will need to sign off on the spec before implementation begins, or will have to sign off on the code before it lands, then you should make them the Approver.

An explanation of the specification data and approval process is at [SpecLifeCycle](#).

You should update the wiki page to include a URL for the Launchpad entry for the spec. This allows you to jump quickly between the wiki spec and its Launchpad entry.

Do not set a priority for your own specifications; this will be determined by the project management teams. If this feature is very important to you, you can help by working on its implementation.

3. **Propose the spec for discussion at a meeting.** Launchpad keeps track of meetings held by the Launchpad and Ubuntu teams. Currently, the only meeting registered is [UbuntuBelowZero](#). You can propose your spec for that meeting agenda using the "Add to Meeting" menu item (top right) on the spec page in Launchpad. Proposals are evaluated the management team; if they can be adequately discussed at the meeting, they will be accepted.
4. **Gather a community around your specification.** Don't assume that people will find it. Announce your spec on ubuntu-devel@lists.ubuntu.com and on #ubuntu-devel. Get people to subscribe to the spec as interested parties. If you have someone who wants to implement it, make them the assignee of the spec. NB, don't make someone the assignee unless they have agreed or you are funding them in some way. The better the community you can build around your idea, the greater the chances that it will be implemented soon, and well.
5. **Work to get approval for the spec.** When your spec is first registered it will be in the 'Brain dump' state. [SpecLifeCycle](#) describes the process through which the specification goes through before being approved.
6. **Register spec dependencies.** The specification tracker allows you to point to any other specifications on which your spec depends: if it only makes sense to implement B after A, you should say so in the Launchpad page for your spec. This is really useful when planning and prioritising the development team's efforts.
7. **Start implementation of the feature.** You should use Bazaar, publish your branches, and register them in Launchpad. In due course we will support


linking of the branches directly to the specifications, so that people can find your code very easily.

CONCLUSIONS

Features that are proposed, caucused, discussed and implemented openly and transparently have the best chance of making it into Ubuntu and into Launchpad. Use this process to steer an idea that you have all the way to delivery. Bear in mind that you can't set priorities or goals for anybody else unless they agree to it, or unless they report to you, or you are willing to fund it. To make something happen, START it and try to gather people around you to make it go faster. Be willing to do it yourself, all the way, if necessary.

(dernière édition le 2005-10-28 16:57:25 par JaneSilber)

SPECSPEC¹⁷⁰

- **Launchpad Entry:**  <https://launchpad.net/distros/ubuntu/+spec/foo>
- **Created:** 2005-10-25
- **Contributors:**
- **Packages affected:**
- **See also:** [SpecTemplate](#)

Summary

This specification describes the way we would like Ubuntu specifications to be written. It takes the form of a specification itself.

Rationale

As we develop new ideas for features in Ubuntu, it's important to be able to communicate them clearly. This serves the purpose of making it clear what the feature is about, and allowing people to evolve an implementation strategy for it.

Publishing this content gives our community a chance to participate in the discussion and design of a feature, and increases the chance that community members will feel confident enough to start work on the implementation of the feature.

A good specification also allows community members who were not physically present at meetings discussing a topic to participate in the implementation of the spec. Bottom line: the better your spec, the better the chances that your ideas will clearly understood by the review team.

¹⁷⁰ Source : <https://wiki.ubuntu.com/SpecSpec>

USE CASES

- Bob is the maintainer for the boot process for Ubuntu. In the Dapper cycle, he would like to work on getting the boot time down to two seconds from boot manager to GDM screen. He creates an entry for the specification in Launchpad, proposes it for the UBZ sprint, and starts writing out a braindump of it in the Ubuntu wiki. Magnus, who is in charge of UBZ scheduling, thinks it sounds fishy but approves it to make sure that the change is discussed and documented properly. He marks it as priority Medium because he isn't sure Bob will have time free for implementing it during Dapper.
- Pedro works on Malone, in Launchpad. Before UBZ, he remembers that the dependency handling in the bug tracker is really not optimal. He writes out a Summary and Rationale in a Launchpad wiki page, registers it as a specification in Launchpad, and suggests it for UBZ. Monica, Launchpad manageress, thinks that this is really not the time to be talking about it and rejects the application for UBZ. He then indicates it for the next conference, UBB, and marks its priority is Low.
- Jason is an Ubuntu and Rosetta user. He has noticed that changes made to translations are making their way into language packs but not to the upstream versions, and adds a specification that describes a way for getting upstream to use language packs. Monica also has a plan for this but hadn't described it in a spec, so she adds it to the UBZ spec list, and adds Carlos, Rosetta maintainer, as drafter for it.

SCOPE

This specification covers feature specifications for Ubuntu and Launchpad. It is not meant as a more general specification format.

DESIGN

A specification should be built with the following considerations:

- The person implementing it may not be the person writing it. It should be clear enough for someone to be able to read it and have a clear path towards implementing it. If it doesn't, it needs more detail.
- That the use cases covered in the specification should be practical situations, not contrived issues.
- Limitations and issues discovered during the creation of a specification should be clearly pointed out so that they can be dealt with explicitly.
- If you don't know enough to be able to competently write a spec, you should either get help or research the problem further. Avoid spending time making up a solution: base yourself on your peers' opinions and prior work.

Specific issues related to particular sections are described further below.

Summary

The summary should not attempt to say **why** the spec is being defined, just **what** is being specified.

Rationale

This should be the description of **why** this spec is being defined.

Scope and Use Cases

While not always required, but in many cases they bring much better clarity to the scope and scale of the specification than could be obtained by talking in abstract terms.

Implementation Plan

This section is usually broken down into subsections, such as the packages being affected, data and system migration where necessary, user interface requirements and pictures (photographs of drawings on paper work well).

IMPLEMENTATION

To implement a specification, the assignee should observe the use cases carefully, and follow the design specified. He should make note of places in which he has strayed from the design section, adding rationale describing why this happened. This is important so that next iterations of this specification (and new specifications that touch upon this subject) can use the specification as a reference.

The implementation is very dependent on the type of feature to be implemented. Refer to the team leader for further suggestions and guidance on this topic.

OUTSTANDING ISSUES

The specification process requires experienced people to drive it. More documentation on the process should be produced.

The drafting of a specification requires english skills and a very good understanding of the problem. It must also describe things to an extent that someone else could implement. This is a difficult set of conditions to ensure throughout all the specifications added.

There is a lot of difficulty in gardening obsolete, unwanted and abandoned specifications in the Wiki.

BOF AGENDA AND DISCUSSION

We'll have a first public session on this on the first Monday in UBZ.

CategorySpec

(dernière édition le 2006-05-25 09:32:07 par Jamespharaoh)

SPECLIFECYCLE¹⁷¹

The Specification Tracker in Launchpad helps enforce a structured process for specification discussion, prioritisation and approval. This page defines the key data

¹⁷¹ Source : <https://wiki.ubuntu.com/SpecLifeCycle>

elements associated with a specification, and describes the process for BOF scheduling and spec approval.

Specification Basics

Each registered specification has the following data elements.

- Title: the person who registers the spec defines this. It will also be used in a URL.
- Summary: a brief description of the goal of the specification
- Product: specs will be attached to Ubuntu or a Launchpad product. Kubuntu and Edubuntu specs should be linked to Ubuntu.
- Priority: Initially this should be unassigned. The product manager(s) will assign a priority as one of
 - Essential: Features which must be implemented.
 - High: Features which are very important and will have resources dedicated to them.
 - Medium: Features which we hope to implement.
 - Low: Features which would likely be included in the system if completed, but which will only have resources assigned if available. Note that sometimes a spec may be low priority at this time, but increase in priority for a later release.
 - Not For Us: Specs in this category may still be a good idea, but are not appropriate for this product.
- Registered date: set automatically by the system

Roles

- Registrant: the person who registered the spec in Launchpad
- Assignee: the person responsible with primary implementation responsibility
- Drafter: the person responsible for completing the specification and getting it to Approved status
- Approver: the person with authority to give approval to a spec
- Subscribers: a set of people who are interested in the spec, would like to contribute to it, attend BOFs to discuss it

Workflow Elements

- Status: Initially this will be Braindump. The spec generally progress through the following states:
 - Braindump: an initial, rough description of the goal, issues, implementation plan
 - Drafting: a work in progress
 - Pending Review: ready for review by a member of the review team for readability and completeness. See [SpecSpec](#) for guidelines on how to pass review
 - Pending Approval: a reviewed spec which is ready for final approval
 - Approved: a golden spec
 - Obsolete: a spec which is no longer relevant

- Superseded: a spec which is no longer relevant and has been replaced with a newer one
 - Informational: a spec which is simply documentation, not for implementation
- Meetings: a sprint or meeting at which the specification will be discussed. In order to get a spec discussed at the meeting, you must add it to the agenda. The meeting organisers will then review the submission mark the spec as:
 - Submitted: under consideration
 - Accepted: spec is appropriate for discussion at the meeting
 - Declined: spec is not appropriate for discussion at the meeting. Note that this is a not a reflection of the validity or priority of a spec. It may be declined simply because the people required for meaningful discussion are not attending the meeting, or because it isn't appropriate for the current development cycle.
- Blockers/Dependents: specs depend on each other. A specification *blocks* another if it must be implemented first. The spec to be implemented later *depends* on the first.
- Needs discussion: A flag to indicate that further discussion BOFs should be scheduled.

Approval Process

The following table reflects the typical approval workflow for a spec.

<i>Who</i>	<i>Initial Status</i>	<i>New Status</i>
Registrant	-	Braindump, Drafting
Drafter	Braindump	Drafting
Drafter	Drafting	Pending Review
Review team	Pending review	Pending approval, Drafting, Braindump
Approver	Pending approval	Approved, Drafting, Braindump

A team of people will be identified as *reviewers* for specs. Reviewers will read the spec for completeness and readability. They will ensure that only specs of sufficient maturity will be presented to the approver. The *approver* will read the spec for technical sanity, implementation details and give the final approval of the specification.

Schedule Process

1. Specs are registered. In order to have it considered by the schedule, it must be added to the meeting.
2. Meeting organisers will accept or decline the spec for the meeting. Normally a priority will be assigned at this time.
3. An automated scheduling system is used to create the BOF schedule. The scheduler takes into account

- Key roles: it will require that the drafter and the assignee attend the session. It will try to get all subscribers to at least one discussion on the topic.
 - Drafting sessions: it will explicitly schedule time for writing up the discussion and decisions taken at the discussion session
 - Priorities: Essential and high priority specs will be scheduled first, followed by medium and low priorities. Sessions for other specs will not be scheduled
 - Available people: names of people who are not assigned in a given time slot will be used to identify lower priority specs which can be discussed
 - Pre-booked slots: sessions can be pre-defined and given to the scheduler as input. This is how to handle allhands sessions, or key meetings which need to take place but are not tied to a given spec. At each meeting the process for requesting one of these sessions will be announced.
4. The schedule will be re-run on a regular basis (e.g., nightly). If a topic needs further discussion, make sure that the *Needs discussion* flag is checked. If this is not checked, the scheduler will not consider this topic.

Implementation Process

The specs continue to be critical throughout the development cycle. The Spec Tracker includes functionality to link a spec to a particular release, or a particular milestone. These flags will be used by the project managers in scheduling work through the course of the development cycle.

(dernière édition le 2005-10-28 16:34:50 par JaneSilber)

POSTFACE

C'est avec beaucoup d'émotion que nous écrivons ces dernières lignes, car elles marquent une étape importante avec la fin de nos études de MBA-Recherche et le dépôt de ce mémoire, c'est le début d'un nouveau cycle de recherche – de doctorat et la fin de cette recherche dans la communauté Ubuntu.

Il fut exaltant, au cours de toute la période de recherche de ce mémoire, de voir se construire petit à petit un système d'exploitation : c'est un projet plein de vie, une conception perpétuelle, des développeurs enthousiastes, des débats et des choix parfois frustrants, parfois réjouissants, toujours intéressants. Le libre est un chaudron géant bouillonnant d'idées, d'idées se concrétisant par l'ajout de nouvelles fonctions, la mise à jour de logiciel innovant, etc. À l'instar de Raymond, nous avons pu constater que

Chaque semaine les concepteurs de logiciels libres inventent et utilisent plus de nouvelles manières de faire les choses que l'industrie propriétaire ne peut le faire en six mois. La raison est que c'est un processus qui comporte moins de tension — lorsque vous voulez tester quelque chose, vous n'avez pas à discuter avec un patron ou risquer de vous faire sacquer parce que votre entreprise pense que c'est trop perturbateur par rapport à leur ligne de produits existants. Vous le faites un point c'est tout.

La liberté est l'oxygène de l'innovation, pas son ennemi¹⁷².

Nous avons observé la communauté dans son fonctionnement quotidien, et nous en avons tiré des enseignements en termes de gestion des connaissances.

Aussi, nous espérons que le libre aura encore de beaux jours devant lui, qu'il saura séduire de plus en plus le monde de l'entreprise, l'administration publique et les

¹⁷² Eric Raymond, « Nous n'avons plus besoin de la GPL », interview sur onlamp.com, traduit sur <http://www.framasoft.net/article4128.html>

groupes communautaires, ce qu'il fait aujourd'hui et fera sûrement demain, et qu'il saura s'étendre à d'autres domaines de la connaissance. Car, outre l'aspect d'efficacité économique du logiciel libre, il y a derrière un ensemble de valeurs que nous apprécions, et jugeons nobles.